Minicurso – NFe no Delphi XE 2: Criação do DANFE

Clube Delphi

Desenvolvimento móvel
Uma solução Android completa com RAD Studio

ITIL
Entenda melhor esse padrão de qualidade internacional

Ano 12 :: Edição 141 :: R$ 11,90

DevMedia

PLATAFORMA ASP.NET 4

Entenda e aplique com Prism XE 2

Herança aplicada
Utilizando herança em suas aplicações

ER/Studio Data Architect XE 2
Modelagem/Documentação de dados de forma prática

ISSN 1517990-7

Somos tão apaixonados por tecnologia que o nome da empresa

diz tudo.

Porta 80 é o melhor que a Internet pode oferecer para sua empresa.

Já completamos 8 anos e estamos a caminho dos 80, junto com nossos clientes.

Adoramos tecnologia.
Somos uma equipe composta de gente que entende e gosta do que faz,

assim como você.

Estrutura

100% NACIONAL. Servidores de primeira linha, links de alta capacidade.

Suporte diferenciado

Treinamos nossa equipe para fazer mais e melhor. Muito além do esperado.

Serviços

Oferecemos a tecnologia mais moderna, serviços diferenciados e antenados com as suas necessidades.

1-to-1

Conhecemos nossos clientes. Atendemos cada necessidade de forma única.
Conheça!

Porta 80
WEB HOSTING

Hospedagem | Cloud Computing | Dedicados | VoIP | Ecommerce | Aplicações | Streaming | Email corporativo

porta80.com.br | comercial@porta80.com.br | twitter.com/porta80

SP 4063-8616 | RJ 4063-5092 | MG 4063-8120 | DF 4063-7486

Sumário

Delphi

WEB

06 – ASP.NET 4 no Prism XE 2
[*Fabrício Hissao Kawata*]

Banco de Dados

14 – ER/Studio Data Architect XE 2
[*Fabrício Hissao Kawata*]

Minicurso

22 – NF2 no Delphi XE 2 – Final
[*Giuliano Scombatti Pinto*]

Minicurso

31 – Desenvolvimento Android – Parte 1
[*Giuliano Scombatti Pinto*]

Easy

38 – Herança aplicada
[*Giuliano Scombatti Pinto*]

Boas Práticas

Boas Práticas

45 – ITIL
[*Paulo Quicoli*]

Olá, eu sou o **DevMan**! Desta página em diante, eu estarei lhe ajudando a compreender com ainda mais facilidade o conteúdo desta edição. Será um prazer contar com sua companhia! Confira abaixo o que teremos nesta revista:

Dê seu feedback sobre esta edição!

A ClubeDelphi tem que ser feita ao seu gosto. Para isso, precisamos saber o que você, leitor, acha da revista!

Dê seu voto sobre esta edição, artigo por artigo, através do link:

www.devmedia.com.br/clubedelphi/feedback

RENOVE JÁ!

Sua assinatura pode estar acabando

Renovando a assinatura de sua revista favorita você ganha brindes e descontos exclusivos.

www.devmedia.com.br/renovacao ou (21)3382-5038

DevMedia

Clube Delphi

Ano 12 - 141ª Edição 2012 - ISSN 1517990-7 Impresso no Brasil

Corpo Editorial

Editor Geral

Paulo Quicoli
pauloquicoli@gmail.com

Editor Técnico

Daniel Sobrinho Laporte
daniel.laporte@gmail.com

Equipe Editorial

Guinther Pauli
guintherpauli@gmail.com

Fabrício Hissao Kawata
fabricio.kawata@bol.com.br

Giuliano Scombatti Pinto
giuliano@sygnux.com.br

Jornalista Responsável

Kaline Dolabella - JP24185

Atendimento ao Leitor

A DevMedia conta com um departamento exclusivo para o atendimento ao leitor. Se você tiver algum problema no recebimento do seu exemplar ou precisar de algum esclarecimento sobre assinaturas, exemplares anteriores, endereço de bancas de jornal, entre outros, entre em contato com:

www.devmedia.com.br/mancad
(21) 3382-5038

Publicidade

Para informações sobre veiculação de anúncio na revista ou no site e para fechar parcerias ou ações específicas de marketing com a DevMedia, entre em contato com:

publicidade@devmedia.com.br

Distribuição

FC Comercial e Distribuidora S.A
Rua Teodoro da Silva, 907
Grajaú - RJ - 206563-900

Fale com o Editor!

É muito importante para a equipe saber o que você está achando da revista: que tipo de artigo você gostaria de ler, que artigo você mais gostou e qual artigo você menos gostou. Fique a vontade para entrar em contato com os editores e dar a sua sugestão!

Se você estiver interessado em publicar um artigo na revista ou no site ClubeDelphi, entre em contato com os editores, informando o título e mini-resumo do tema que você gostaria de publicar:

Paulo Quicoli - Editor da Revista
pauloquicoli@gmail.com

EDITORIAL

Sem dúvida a informatização está passando por uma transformação, a chamada era pós-PC. Ela não quer dizer que todos deixaremos de utilizar PCs e vamos realizar nossas tarefas em tablets e smartphones. Acredito que todo novo dispositivo com poder de processamento de dados e conectividade disponível que surgir, será utilizado como um braço a mais de um aplicativo que roda em algum desktop, isto, comercialmente falando. Já é realidade termos software de frente de vendas que são executados em dispositivos móveis, mas seu backend é o servidor local da empresa, que processa toda informação coletada. Toda essa mudança força, empurra, o desenvolvimento de novas tecnologias, sistemas operacionais e linguagens de programação. O desenvolvedor atual que quer manter-se no mercado não pode mais associar sua profissão a um único produto, por exemplo: sou desenvolvedor Delphi. Na verdade desenvolvedor é desenvolvedor, não importa sua linguagem. O que quero dizer com isto é que existem ferramentas específicas para cada ambiente. A Embacadero reconhecendo isto criou o RadPHP como recursos que permitem o desenvolvimento de para Android. Assim, podemos aprendê-la e criar aplicativos que acessam uma fonte de dados disponibilizada na web e testar em nosso próprio smartphone. Uma solução hoje de mobilidade envolve mais de uma tecnologia e o Giuliano inicia com seu mini-curso. A solução proposta envolve o uso de PHP para a criação da aplicação cliente e Delphi (DataSnap) para a criação da aplicação servidora. Esse é um cenário comum de integração de tecnologias. Nesse tema WEB ainda, o Fabrício mostra o lado .NET com o Prism XE 2 e o ASP.NET 4, seus recursos e funcionalidades. Vejam, este é um caminho sem volta, mobilidade, dados disponíveis na WEB.

Mas, não importa se o aplicativo é online ou local, ainda temos bancos de dados por traz de tudo isso. Assim, é muito importante realizar uma modelagem e documentos do modelo de dados. O desenvolvedor Embarcadero tem soluções disponíveis para isso também. O Fabrício mostra como utilizar o ER/Studio Data Architect para modelagem de banco de dados.

Para a seção Easy, destinada a quem está ingressando no mundo do desenvolvimento Delphi, temos um artigo do Giuliano que mostra como utilizar de forma eficiente o recurso de herança da orientação a objetos. O Giuliano também encerra seu mini-curso sobre NFe com FireMonkey, mostrando como realizar a impressão do DANFE e eu, para finalizar, trago um artigo sobre ITIL. Se você está pensando em abrir sua empresa, ou já é dono do seu próprio negócio, veja neste artigo alguns conceitos dessa metodologia que pode fazer a diferença em sua organização.

Boa leitura!

Paulo Quicoli
pauloquicoli@gmail.com
Editor Geral
quicoli.wordpress.com
twitter.com/pauloquicoli

MVP

Assine agora e tenha acesso a todo o conteúdo da DevMedia:
www.devmedia.com.br/mvp

DevMedia

Dificuldades em encontrar conteúdo sobre .Net para iniciantes?

A DevMedia simplifica pra você!

easy .net magazine

Easy .Net Magazine:

A primeira revista digital feita especialmente para quem está dando os primeiros passos em .Net!

Você encontrará exemplos e modelos fáceis e práticos, noções introdutórias e conceitos básicos.

Conheça as vantagens:

- Vídeos e apresentações "embutidos" nas páginas;
- Ilustrações, diagramas e modelos apresentados em PowerPoint;
- Entrevistas em vídeo;
- Códigos-fonte que podem ser copiados com CTRL+C/CTRL+V.

A Easy.Net não é feita em PDF ou Flash!
Ela utiliza um aplicativo 100% online.

DevMedia group

Acesse agora a edição Nº 1 totalmente grátis e comprove

www.devmedia.com.br/easynet

Easy.Net Magazine, seu curso inicial de .Net!

ASP.NET 4 no Prism

Desenvolvimento de aplicações p

A década de 90 marcou um dos primeiros anos de vida da Web. Ainda naquela época o modelo de programação utilizado para se construir um simples site, envolvia basicamente a linguagem de marcação HTML e, eventualmente, objetos JavaScript. As páginas Web tinham então tanto seu código quanto a definição de sua interface visual escritos de forma manual, o que requeria um foco trabalhoso bem como um alto grau de conhecimento. Já por volta de 2001, com o intuito de melhorar as práticas de desenvolvimento da atualidade, a Microsoft dava início ao ASP.NET. Dentre suas características, este novo Framework unia itens relacionados a duas tecnologias da época: a melhor desenvoltura do então ASP com o modelo RAD (Rapid Application Development) visto no Visual Basic. Por sinal, este mesmo modelo de desenvolvimento rápido de aplicações é o que consagrou o Delphi, e que reflete a facilidade em se construir uma aplicação real em um curto espaço de tempo.

Em vista disso, sua concepção teve a intenção principal de possibilitar aos desenvolvedores a construção de aplicações Web de uma forma fortemente "tipada" e orientada a objetos. A ideia era tornar o desenvolvimento Web o mais prático possível, a ponto de, por consequência, minimizar o choque inevitável estabelecido pela mudança de plataforma (Desktop para Web), privilegiando assim principalmente os profissionais provenientes do desenvolvimento Win32.

Do ponto de vista evolutivo do Framework, a partir de sua primeira versão algumas das características apresentadas se tornaram essenciais ao modelo proposto, sendo então mantidas. Ao mesmo tempo, outras novas foram sendo agregadas ao longo do tempo. Dentre suas principais características presentes desde a versão 1.x estão:

- Em uma aplicação Web ASP.NET, as páginas são essencialmente codificadas com o uso de linguagens de programação tradicionais suportadas pelo .NET, tais como C#, VB.NET e Oxygene (mais recentemente);

Resumo DevMan

De que trata o artigo:

O presente artigo aborda de uma maneira geral o desenvolvimento Web no Prism XE2, através do uso da plataforma ASP.NET, que se mostra como sendo uma das principais opções para a construção de aplicações deste segmento (Web). A introdução no assunto se dá primeiramente por situar o leitor sobre o estágio atual do Framework, expondo algumas vertentes de sua gama de possibilidades, originadas desde sua primeira versão até sua atual, o ASP.NET 4.0. Com isso, uma base de conhecimento é estabelecida, favorecendo futuras abordagens sobre a tecnologia, de forma que o trabalho com a mesma se torne tão natural quanto o desenvolvimento Desktop no Delphi tradicional.

Em que situação o tema é útil:

Grande parte dos programadores Delphi tem no desenvolvimento Desktop sua principal, e talvez única, opção de trabalho. No entanto, com a fluidez com que o mundo corporativo seguiu até o momento atual fez com que novas necessidades surgissem ao longo do tempo. Dentre estas se encontra o estabelecimento de aplicações voltadas para Web, que expõem um paradigma diferente do habitual cenário Desktop. Para a construção destas, a comunidade Delphi dispõe de uma ferramenta apreciada de longe e pouco explorada por muitos, o Prism, que recentemente também ganhou sua versão XE2, da mesma forma que o Delphi. Ele se integra ao Visual Studio, tirando proveito de todos os recursos do Microsoft .NET Framework e aumentando a viabilidade em se adotar o ASP.NET para o desenvolvimento de aplicações para a Web.

ASP.NET 4 no Prism XE2:

Com a necessidade de se construir aplicações para Web, o programador Delphi sai de sua zona de conforto, representado pelo desenvolvimento Desktop, para adentrar a um novo cenário completamente diferente. Assim sendo, a primeira grande decisão é optar por uma, entre as várias vertentes disponíveis atualmente, voltadas para este determinado segmento. Somente no nicho Embarcadero, muitas são as opções, tais como o PHP, através do RadPHP XE2, o IntraWeb, disponível no próprio Delphi, além do ASP.NET (WebForms e MVC) e Silverlight no Prism XE2. Especificamente falando sobre ASP.NET, é de longa data considerada uma plataforma madura, sendo então uma opção segura para se adotar. Ela é toda baseada no .NET Framework, herdando suas características e possibilitando que as aplicações possam ser escritas em diversas linguagens de programação. Entre elas, a Oxygene, a linguagem utilizada no Prism XE2 e que é também baseada em Object Pascal, a mesma do Delphi. Este simples detalhe faz com que desenvolvedores Delphi que ingressem no desenvolvimento Web com ASP.NET, através do Prism, tenham uma redução na curva do aprendizado, em vista do reaproveitamento de conhecimento da linguagem. Este artigo então tem o objetivo de expor um aspecto geral deste tipo de desenvolvimento, e o contexto da plataforma até sua versão mais atual, a de número 4. Por conseguinte, isto dará margem para o devido aprofundamento em textos posteriores.

XE2
ara Web no IDE da Embarcadero

Fabrício Hissao Kawata

• Ideal separação entre presentation logic (camada de apresentação) e business logic (camada de negócios), o que significa dizer que durante o desenvolvimento o código de apresentação (HTML) não se mistura ao código de negócios (C#, Oxygene). Tudo isto proporcionado por um formato de modelo que veio a ser conhecido como code-behind;
• O ASP.NET provê uma gama de controles denominados Web Controls, que permitem que se construam interfaces gráficas (GUIs) para aplicações Web, da mesma forma que em aplicações Desktop, numa tentativa de aproximar ao máximo esses dois mundos;
• Nativamente, o ASP.NET utiliza um mecanismo especial nos formulários de suas páginas denominado ViewState, afim de manter o estado de seus controles durante as requisições, uma vez que o padrão Web é stateless. Com isso, consegue-se um efeito semelhante ao das aplicações Desktop;
• Uma aplicação ASP.NET é compilada em um assembly .NET, que poderá então ser gerido pela CLR (Common Language Runtime), além de otimizar sua execução;
• Em termos de configuração, uma ASP.NET Web Application pode ser nativamente configurada no ambiente Windows, diretamente no servidor Web IIS (Internet Information Services) ou ainda através de um arquivo de configuração específico próprio (Web.config).

Nota do DevMan

CLR é a sigla para Common Language Runtime. Ela é tida no popular, como sendo o coração do .NET, podendo ser considerada uma espécie de máquina virtual encarregada de executar as aplicações voltadas à plataforma. Graças a ela, as distintas linguagens de programação suportadas pelo .NET, tais como Oxygene e C#, conseguem se interoperar. Para isso, são definidas uma série de especificações que o compilador de cada linguagem deverá seguir, a fim de produzir um código numa linguagem de mais baixo nível (CIL - Common Intermediate Language) que será então interpretado pela plataforma.

Enquanto que para se montar uma típica página HTML seria necessária alguma digitação manual de código HTML, no ASP.NET isto não é necessário. Isto porque, uma das grandes facilidades proporcionadas pelo Framework é a possibilidade de se arrastar e soltar seus controles em tela, da mesma forma que em uma aplicação VCL Forms no Delphi. Feito isso, esses controles são então renderizados no servidor, que emite de volta ao cliente (Browser) uma resposta contendo as tags HTML condizentes aos controles. Exemplificando, um simples controle Label do ASP.NET, quando arrastado ao WebForm, produz um código semelhante ao a seguir:

```
<asp:Label ID="lblLabelTeste" runat="server" BackColor="Yellow"
      Font-Bold="True" Text="Label Teste"></asp:Label>
```

Este código, uma vez renderizado, produz a saída a seguir, que se trata tão somente de um HTML padrão. Esta marcação poderá então ser processada por qualquer navegador Web (Internet Explorer, Mozilla FireFox, Google Chrome, etc), inclusive em dispositivos móveis que os suportem.

```
<span id="lblLabelTeste"
   style="background-color:Yellow;font-weight:bold;">Label Teste</span>
```

Atualmente o ASP.NET é considerado, de forma unânime, uma plataforma madura o suficiente para proporcionar a estabilidade necessária para tornar o desenvolvimento Web altamente produtivo. Até chegar a este ponto, o Framework foi constantemente melhorado e refinado, incluindo extensibilidades inéditas ao seu início, como também manteve pontos presentes desde sua origem. Dito isto e sem querer fazer um apanhado detalhado de todo o histórico de evolução da plataforma até o momento, é importante situar o leitor sobre algumas das nuances pontuais que aconteceram durante sua jornada.

Breve histórico de evolução

O ASP.NET 1.x mostrava-se então muito bem consolidado para a construção de aplicações Web, provendo através de seus recursos o dinamismo necessário às suas páginas e seu desenvolvimento. Em sua versão de número 2.0, a intenção era poder construir sites ainda mais dinâmicos e ricos, além de agilizar ainda mais o processo de desenvolvimento. Exemplo disso foi a introdução de um servidor Web interno, direcionado exclusivamente ao período de construção da aplicação, isentando do desenvolvedor a necessidade de possuir o Internet Information Services instalado em sua máquina de trabalho. Além disso, aumentou-se consideravelmente o número de Web Controls disponíveis para a plataforma, que abrangiam áreas como segurança, navegabilidade e ligação de dados (Data Binding).

Também foram introduzidos nesta versão os tradicionais recursos de Master Pages e Themes. A primeira, como sua própria nomeação sugere, é uma página que irá servir de base para outras páginas da aplicação, definindo uma aparência e um comportamento padrão. Funciona como uma espécie de página "modelo". Dessa forma, consegue-se definir um padrão que será comum

a um conjunto de páginas do Web Site, tal como um mesmo cabeçalho, rodapé, menu de navegação além de outros elementos. Em vista disso, na medida em que seja necessária uma eventual reformulação do aspecto geral da aplicação, somente a Master Page será mudada, incidindo automaticamente nas páginas que a utilizam. Já o recurso de Themes é uma forma declarativa para se alterar a aparência das páginas e controles de aplicação Web no próprio servidor. Para isso, são utilizados diversos temas, onde cada qual detém uma coleção de configurações de propriedades que irão refletir sobre o visual das páginas e seus controles. Tais configurações podem estar divididas em uma série de elementos, tais como arquivos de imagens, recursos e estilos (CSS), definidos em diretórios especiais.

Na mesma medida, ainda a fim de prover páginas mais ricas e interativas, aparecia o suporte a Web Parts, caracterizado por um conjunto específico de controles. Em suma, através do uso destes, provia-se ao usuário final a capacidade de personalização de uma página Web, desde o simples arrastar-e-soltar suas partes até estilizar seus elementos. Em outras palavras, a possibilidade de modificar aparência, conteúdo e comportamento de uma página, através de uma interação direta com o navegador, armazenando tais configurações para uso posterior.

O histórico de versões do ASP.NET se correlaciona com o do .NET Framework. Sem deixar de mencionar, foi ainda na versão 3.0 deste último que foram inseridas tecnologias como WPF (Windows Presentation Foundation), WF (Windows Workflow Foundation), WCF (Windows Communication Foundation) e CardSpace (Windows CardSpace), e que posteriormente vieram a se consolidar ao longo do tempo. Voltando ao ASP.NET, sua versão 3.5 trouxe, entre outras coisas, o suporte ao novo modelo de programação LINQ, que permitia uma nova forma de se realizar consultas em variadas fontes de dados através do uso de linguagem de programação. Além disso, foi integrado o devido suporte ao desenvolvimento baseado em Ajax, que essencialmente provia a atualização parcial de uma página Web ao invés do todo. Tal suporte se refletiu então em um novo conjunto de controles exclusivos ao contexto, responsáveis por fazer todo o trabalho pesado de fundo envolvendo JavaScript. Surgiu também o apoio à construção de aplicações Web orientada a dados através do ASP.NET Dynamic Data. Em suma, isso significava dizer que a partir de uma simples tabela de banco de dados, seus dados poderiam ser renderizados de forma direta e automática para HTML. Por essa forma ágil de desenvolvimento baseada em dados fez com que o ASP.NET Dynamic Data fosse comparado, até certo ponto, ao desenvolvimento com Ruby on Rails.

Finalmente, ao falar sobre o ASP.NET 4, é possível citar também o suporte oficial a um novo modelo de projetos denominado ASP.NET MVC. Isto porque, por um longo tempo o termo ASP.NET era sinônimo de aplicativos para Web construído com o tradicional modelo Web Forms. Então, em termos práticos, o que se viu foi o ASP.NET MVC inserindo um novo tipo de abordagem, que vai de encontro ao modelo tradicional de desenvolvimento presente até então (o Web Forms), mudando radicalmente a forma de programar. Ele faz uso do famoso Design Pattern MVC (Model-View-Controller) para a construção das aplicações, dividindo-as em camadas levemente interdependentes. Alguns até se arriscam a dizer que a abordagem Web Forms é voltada ao profissional provindo do desenvolvimento Desktop, enquanto que o ASP.NET MVC é direcionado à essência real do desenvolvimento Web e, portanto, para este tipo de profissional. Além disso, outra incorporação significativa nesta versão foi a adição de um novo controle voltado especificamente para a inclusão de gráficos de análises em páginas ASP.NET, o ASP.NET Chart.

Nota do DevMan

Asynchronous Javascript and XML, ou simplesmente AJAX, pode ser conceituado como sendo uma metodologia com o intuito essencial de aumentar a interatividade das páginas Web com o usuário final. Em vista disso, faz uso de tecnologias como XML e JavaScript para proporcionar solicitações assíncronas de informações, reduzindo assim o tráfego de dados pela rede e, consequentemente, otimizando o tempo de resposta de uma solicitação.

Indo um pouco além, a seguir serão abordados alguns destes e de outros novos recursos, principalmente voltado à parte de controles, introduzidos na versão 4 do ASP.NET. Apenas para ficar claro, o intuito não é fazer uma abordagem completa das novidades que esta versão da plataforma trouxe, e sim apresentar algumas de suas nuances e, consequentemente, familiarizar ainda mais o desenvolvedor Delphi emergente na tecnologia ASP.NET e no desenvolvimento Web.

ClientIDMode

Quanto se adiciona um simples controle ASP.NET em uma página, em seu código de definição é automaticamente definida sua propriedade runat com o valor server. Tal indicativo estabelece que este controle será processado no servidor, o que essencialmente o difere de um controle HTML tradicional. Olhando para o contexto do desenvolvimento Web, há várias situações onde é necessário ter acesso a um determinado controle HTML presente na página. Nestes casos, é preciso poder identificar o controle a meio a tantos outros objetos que também fazem parte da página, utilizando para isso o seu próprio identificador (id). Um exemplo clássico desse uso seria em funções JavaScript, que são definidas no lado cliente e, portanto, necessitam do identificador do controle HTML já renderizado na página. O método a seguir, GetElementId, é prova disso. Através dele, é possível obter determinado elemento de página, através de seu identificador.

```
var element = document.getElementById("lblLabelTeste");
```

No entanto, na prática, dependendo do cenário pode haver diferenças entre o id do controle ASP.NET presente na página, com o id presente em seu código HTML que é gerado e retornado pelo servidor. Isso pode ser facilmente percebido com a utilização de Master Pages. A partir de um projeto ASP.NET no Prism XE2, e

tomando como base um controle Label cujo ID seja "lblLabelTeste", este é inserido em dois lugares: em uma página Web Forms simples e em outra baseada em uma Master Page. Logo, em termos do HTML retornado referente ao controle, se teria os seguintes resultados:

```
//Sem uso de Master Pages
<span id="lblLabelTeste">Label Teste</span>

//Usando Master Pages
<span id="ContentPlaceHolder1_lblLabelTeste">Label Teste</span>
```

Via de regra, o atributo id do elemento HTML é definido com o valor do ClientID do controle ASP.NET. Conforme pode se subentender, esta propriedade somente leitura retorna um identificador para o controle que, como visto, pode variar de caso para caso. Em vista disso, uma das novidades apresentadas pelo ASP.NET 4, é o atributo ClientIDMode, que é utilizado então para a obtenção e definição do algoritmo a ser utilizado na geração do valor do ClientID. Atualmente são quatro as opções de algoritmos possíveis:

- Inherit: algoritmo que determina que o controle herde o valor para ClientIDMode de seu controle pai. É o valor padrão para o ClientIDMode dos controles no ASP.NET 4;
- Predictable: algoritmo voltado a controles envolvendo ligação de dados. Como base, o valor do ClientID de um controle é determinado pela concatenação dos ID's de seus containers (caso hajam) e seu próprio ID. Ainda em casos especiais, onde a ligação de dados produza várias linhas contendo um mesmo tipo de controle, pode ser então especificado um sufixo para distinguir o valor de cada ClientID. Isto é feito pelo uso da propriedade ClientIDRowSuffix. No ASP.NET 4, Predictable é o valor padrão para o ClientIDMode de uma página;
- Static: este algoritmo determina que o valor do ClientID deva ser exatamente igual ao ID do controle;
- AutoID: é o algoritmo utilizado pelas versões anteriores ao ASP.NET 4, em que o valor do ClientID é gerado concatenando-se os ID's de cada container pai e o ID do próprio controle.

ViewStateMode

Conforme já mencionado, o ASP.NET utiliza-se de um recurso denominado ViewState para preservar os valores de uma página e de seus controles durante as requisições da mesma. No HTML produzido, essas informações ficam armazenadas em um controle do tipo Hidden Field. Justamente por isso, caso uma quantidade excessiva de informações forem mantidas neste campo, incidirá diretamente na performance da página, degradando sua velocidade. Todavia, sem o emprego desta metodologia, os valores inseridos no contexto da página seriam simplesmente perdidos durante um Postback. Em outras palavras, o valor digitado em um TextBox, por exemplo, seria perdido na medida em que ocorre o processo de ida e volta da página ao servidor. Elucidando um pouco do que foi dito até agora, a seguir é mostrado um trecho de código HTML produzido a partir de uma página ASP.NET, que mostra a declaração do Hidden Field que representa o ViewState da página. Detalhe para o atributo value do elemento, que representa as informações armazenadas no processo, já devidamente serializadas em uma codificação de Strings no formato base64-encoded. Só para citar, esta codificação prevê uma sequência contendo somente os seguintes caracteres contendo +, /, a(A)-z(Z) e 0-9.

```
<div class="aspNetHidden">
<input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE" value="/wEPDwU-
JMjgzMDgzOTgzZGRoJXeI14/zJpmXMuZL0iwqhFmlBm0aHXehTVPp8FpNgQ==" />
</div>
```

Logicamente, o objetivo de se manter os valores dos objetos durante uma requisição depende diretamente da situação e da necessidade de cada projeto. Assim sendo, no ASP.NET 4 foi inserida aos Server Controls uma nova propriedade, chamada ViewStateMode, que proporciona ao desenvolvedor um controle maior sobre o ViewState. Na prática, através de seu uso será possível basicamente habilitar e desabilitar o ViewState de páginas e controles. Consequentemente, abre-se a possibilidade de se realizar outras combinações neste sentido, tal como habilitar o ViewState somente de um controle específico, mesmo com o ViewState da página desabilitado. Os valores possíveis para o ViewStateMode são três: Enabled, Disabled e Inherit.

O primeiro, Enabled, habilita o ViewState de um controle, independente de seu controle pai estar com ViewState habilitado ou não. De forma inversa, Disabled desabilita o ViewtState do controle, novamente independente do que foi definido em seu ancestral. Já Inherit faz com que o valor para o ViewStateMode seja herdado do controle pai.

Em termos de usabilidade, uma observação a ser feita sobre o ViewStateMode, é que o valor definido para ele só será efetivo se a propriedade EnableViewState possuir o valor True. Esta propriedade, aliás, é quem irá determinar o comportamento do ViewState, caso a propriedade ViewStateMode não esteja definida a nível de página ou controle. Da mesma forma, sem deixar de mencionar, o valor padrão do ViewStateMode para uma página é Enabled, enquanto que para os Server Controls é Inherit.

Persisted Selection

Em aplicações Web ASP.NET, ao se trabalhar com acesso a dados, é comum a utilização de controles como GridView e ListView para a listagem e manipulação rápida das informações. Nestes casos, cada linha apresentada pelo controle representa um registro da fonte de dados (Data Source) ao qual está ligada (Ex: uma tabela do banco de dados). Sendo assim, a seleção de uma de suas linhas faz com que se possa efetuar eventuais manipulações sobre aquele determinado registro. Em versões anteriores do ASP.NET, esta seleção de linha era toda baseada num item denominado row index, que nada mais é que o índice da linha. Na prática, apesar de funcional, esta forma de lidar com a linha selecionada causava certos inconvenientes. Um exemplo clássico é quando da utilização de um controle GridView envolvendo paginação de registros. Em casos como este, a seleção de linha efetuada em uma página era mantida ao se navegar para outra página, o que não é desejável na maioria dos casos. Em suma, ao se fazer a seleção do segundo item da página 1, esta seleção se mantinha na página 2 e assim sucessivamente, de forma automática, mesmo sem o usuário ter tido a intenção de efetuar ou manter a seleção na segunda página. Todo este comportamento ocorre pelo fato da seleção ser baseada no índice da linha e não no registro em si.

Este comportamento pode ser visto a seguir, nas **Figuras 1** e **2**, onde esboça-se a seleção de registros sendo mantida entre a navegação pela paginação.

No ASP.NET 4, foi introduzido então, de forma oficial, um novo recurso até então suportado apenas em projetos do ASP.NET Dynamic Data ainda no .NET Framework 3.5 SP1, denominado Persisted Selection. Tal recurso se reflete em uma nova propriedade denominada EnabledPersistedSelection que, uma vez habilitada, faz com que a seleção em controles de dados, tais como o GridView e ListView, seja baseada no row data key, uma espécie de chave que especifica o registro em si. Dessa forma, consegue-se então produzir o efeito desejado para a seleção de registros nestes controles onde há paginação de dados. Isso é demonstrado nas **Figura 3** e **4**, onde é mostrado um comportamento que é tido como o natural, onde a seleção é feita exclusivamente sobre determinado registro. Nas imagens, num primeiro momento é mostrada a seleção do registro desejado na primeira página e, posteriormente, ao avançar para a segunda página, nenhuma linha mostra-se selecionada. Sem deixar de mencionar, para que o recurso funcione adequadamente, é necessário definir um valor à propriedade DataKeyName do controle que, basicamente, identifica o campo chave primária do Data Source utilizado. Uma vez definido, a seleção será então baseada neste

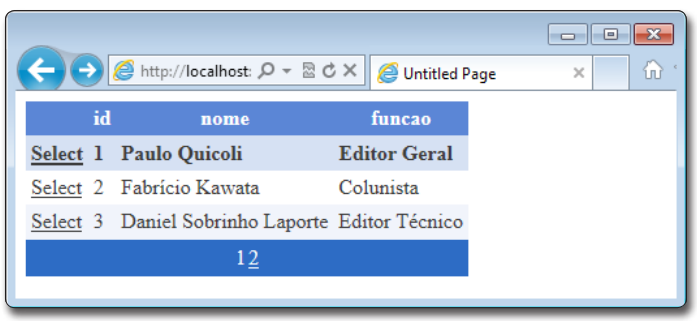


Figura 1. GrideView – Seleção – página 1

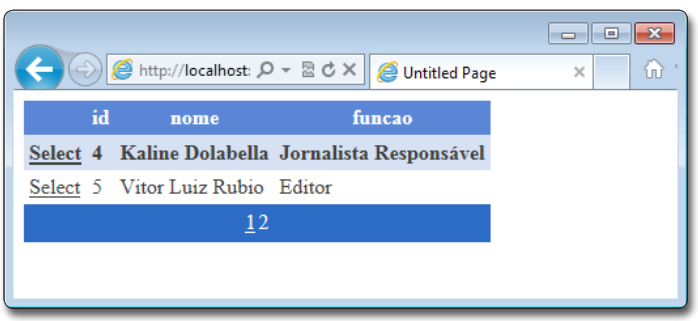


Figura 2. GrideView – Seleção – página 2

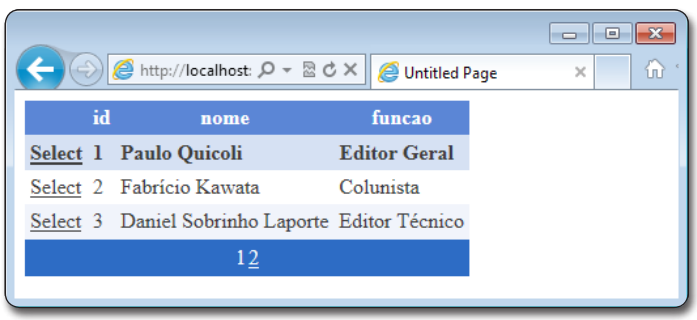


Figura 3. GridView com PersistedSelection habilitado – página 1

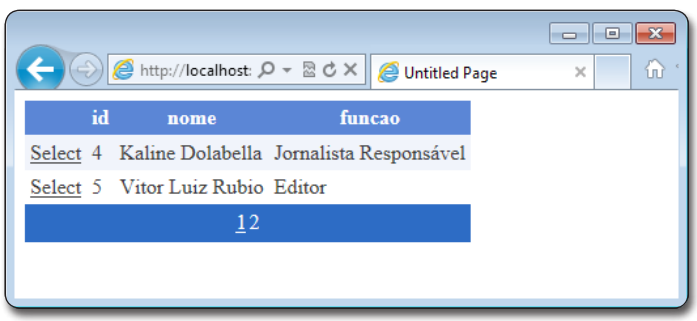


Figura 4. GridView com PersistedSelection habilitado – página 2

campo. Intuitivamente, para este exemplo foi utilizado o campo id. A declaração ASP.NET inicial do controle é mostrada a seguir:

```
<asp:GridView ID="GridView1" runat="server" AllowPaging="True"
        DataKeyNames="id"
        DataSourceID="SqlDataSource1"
        EnablePersistedSelection="True"
        PageSize="3">
```

Melhorias do FormView

O FormView é um controle ASP.NET voltado a dados, mais especificamente para a manipulação das informações da fonte de dados (Data Source) a qual está ligado. Sua característica principal é a de exibir um único registro por vez em tela, permitindo que se definam modelos (Templates) tanto para exibição como edição dos dados. Um exemplo típico de uso é sua combinação com um controle GridView. Neste caso, os registros são listados na grade, enquanto que os detalhes do item selecionado são mostrados no FormView, permitindo se atualizar ou inserir novos registros. A **Figura 5** mostra um exemplo trivial de sua utilização, somente para visualização e atualização de dados. Esta demonstração será usada então para se explicitar uma nova melhoria trazida no ASP.NET 4 para o controle.

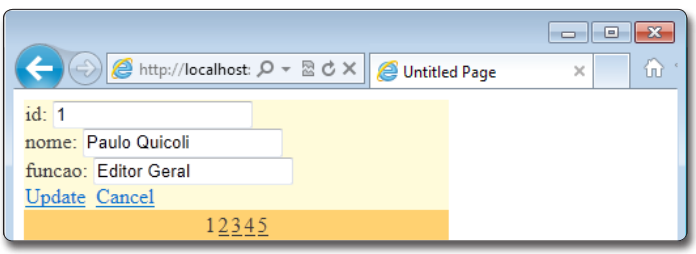


Figura 5. FormView para exibição e edição de dados

Com base na imagem apresentada, um trecho da marcação HTML produzida pelo controle ASP.NET seria:

```
//Código ASP.NET
<asp:FormView ID="FormView1" runat="server" AllowPaging="True"
        DataSourceID="SqlDataSource1" DefaultMode="Edit" CellPadding="4"
        ForeColor="#333333">
...
</asp:FormView>

//Código HTML
<table cellspacing="0" cellpadding="4" id="FormView1"
    style="color:#333333;border-collapse:collapse;">
  <tr style="color:#333333;background-color:#FFFBD6;">
    <td colspan="2">
        id:
        <input name="FormView1$idTextBox" type="text" value="1"
          id="FormView1_idTextBox" />
        <br />
        nome:
        <input name="FormView1$nomeTextBox" type="text"
          value="Paulo Quicoli" id="FormView1_nomeTextBox" />
        <br />
        funcao:
        <input name="FormView1$funcaoTextBox" type="text"
         value="Editor Geral" id="FormView1_funcaoTextBox" />
        <br />
...
</table>
```

Em vista do código apresentado, conclui-se que a utilização de um controle FormView em uma página ASP.NET, terá como resultado final no Browser uma marcação HTML de um elemento Table. Este era o comportamento padrão de renderização do controle até o ASP.NET 4, quando foi então introduzida uma nova propriedade booleana para o controle, denominada RenderOuterTable. Ao ser atribuída como False, ela determina que o FormView não mais seja mais renderizado em um elemento Table. Em substituição, é então utilizado um elemento Div, o que ocasionalmente pode acabar por proporcionar uma maior flexibilidade em termos de estilização do conteúdo com CSS (Cascading Style Sheets). Olhando pelo lado prático, se o mesmo FormView apresentado anteriormente fosse configurado utilizando-se RenderOuterTable, seu HTML resultante seria conforme mostrado a seguir:

```
//código ASP.NET
<asp:FormView ID="FormView1" runat="server" AllowPaging="True"
        DataSourceID="SqlDataSource1" DefaultMode="Edit"
        RenderOuterTable="False">
...
</asp:FormView>

//código HTML
<div>
    id:
    <input name="FormView1$idTextBox" type="text" value="1"
    id="FormView1_idTextBox" />
    <br />
    nome:
    <input name="FormView1$nomeTextBox" type="text"
    value="Paulo Quicoli" id="FormView1_nomeTextBox" />
    <br />
    funcao:
    <input name="FormView1$funcaoTextBox" type="text" value="Editor Geral"
    id="FormView1_funcaoTextBox" />
    <br />
...
</div>
```

Web Standards e Acessibilidade

Em suma, o termo Web Standards é designado para definir os padrões determinados pela W3C (World Wide Web Consortium) para a construção de WebSites. Tais normas estão intimamente ligadas ao código que é utilizado na definição da página Web, tais como a utilização invariável de código HTML e XHTML válidos, bem como o emprego de CSS. O intuito principal destas recomendações é de que sejam desenvolvidos sites genuinamente de boa qualidade e providos de uma devida acessibilidade universal. Isso quer dizer proporcionar uma "Web para todos", tornando a página acessível inclusive para pessoas com algum tipo de deficiência como, por exemplo, a visual.

Olhando pelo lado prático, tomemos como base o trecho de código apresentado na **Listagem 1**, referente a uma hipotética aplicação Web utilizando versões anteriores à 4 do ASP.NET, tais como a 2.0 ou 3.5. Neste simples exemplo demonstra-se o emprego de dois controles, um Label e um TextBox, em determinado Web Form da aplicação. Logo, ambos têm suas propriedades Enabled definidas com o valor False, para que fiquem "desabilitados" em tela.

Listagem 1. Código dos controles ASP.NET de exemplo

```
<asp:Label ID="lblLabelTeste" runat="server" Enabled="False"
Text="LabelTeste"></asp:Label>
<asp:TextBox ID="txtTextBoxTeste" runat="server" Enabled="False">
</asp:TextBox>
```

Uma vez renderizados, tais controles produziriam uma saída HTML conforme a seguir:

```
<span id="lblLabelTeste" disabled="disabled">LabelTeste</span>
<input name="txtTextBoxTeste" type="text" id="txtTextBoxTeste" disabled="disabled" />
```

Assim sendo, falando especificamente das versões anteriores da plataforma, os controles ASP.NET utilizados numa aplicação por vezes não produziam uma marcação HTML e XHTML adequada, não condizentes com os conceitos de Web Standards e Acessibilidade citados. Prova disso é o próprio código HTML de exemplo. Conforme visto, foram produzidos dois elementos HTML, span e input, ambos com um atributo disabled declarado. No entanto, com base no HTML 4.01, este atributo não é válido para elementos que só mostram informações, os chamados display-only, sendo sugeridos somente a elementos de entrada de dados (input). Sendo assim, o emprego do atributo disabled ao elemento span é inválido.

Em função desse tipo de desacordo, a versão 4 do ASP.NET foi concebida visando uma melhoria natural neste aspecto, acabando por eliminar grande parte destes "problemas". Tomando então como base o exemplo apresentado, no ASP.NET 4 os mesmos controles produzem então a seguinte marcação HTML:

```
<span id="lblLabelTeste" class="aspNetDisabled">LabelTeste</span>
<input name="txtTextBoxTeste" type="text" id="txtTextBoxTeste"
disabled="disabled"   class="aspNetDisabled" />
```

Dessa forma, o elemento span ao invés de um atributo disabled, possui agora um atributo class, definido como aspNetDisabled. Tal atributo poderá então ser utilizado, por exemplo, na definição de um estilo (CSS) voltado especificamente para se caracterizar controles "desabilitados".

Conclusão

Utilizar a plataforma ASP.NET para o desenvolvimento Web pode ser a alternativa mais atraente ao profissional provindo do desenvolvimento Desktop, principalmente em razão da utilização de seu tradicional modelo Web Forms. Isto porque, conforme já dito, sua forma básica de trabalho apresenta uma grande proximidade com o Delphi, visto seu poder RAD e o suporte à linguagem Oxygene, que também é baseada em Object Pascal e provida pelo Prism XE2.

Além dessa, já olhando para uma futura maturidade atingida com o desenvolvimento ASP.NET, outra opção de trabalho seria com o também mencionado ASP.NET MVC. Cita-se o termo maturidade, em vista do modelo de programação utilizado pelo Framework ser totalmente diverso ao modelo Web Forms. Tanto é verdade que desde seu surgimento, houve uma preocupação clara em se distinguir cada vertente, ressaltando suas diferenças e, principalmente, salientando a implementação do Pattern MVC no novo Framework.

Agradeço a todos e até a próxima.

Fabrício Hissao Kawata
fabricio.kawata@bol.com.br
Formado em Processamento de Dados pela FATEC-TQ. Atua como Analista Programador Delphi há 6 anos. Atualmente escreve artigos sobre Delphi para revistas especializadas.
Blog: prismsilver.blogspot.com

Embarcadero Prism XE2 – Página oficial do produto
http://www.embarcadero.com/products/prism

Embarcadero Prism XE2 - Trial
https://downloads.embarcadero.com/free/prism

Microsoft ASP.NET - Página oficial
http://www.asp.net

Microsoft ASP.NET MVC - Página oficial
http://www.asp.net/mvc

Dê seu feedback sobre esta edição!

A Clubedelphi tem que ser feita ao seu gosto. Para isso, precisamos saber o que você, leitor, acha da revista!

Dê seu voto sobre este artigo, através do link:

www.devmedia.com.br/clubedelphi/feedback

Seu **conhecimento vale muito** na DevMedia

Publique artigos e vídeos gratuitamente no site e participe do programa **PayContent.**

Sua participação pode ser premiada: quanto mais você postar, mais chances de ganhar.

Para saber mais acesse:
www.devmedia.com.br/paycontent

DevMedia

ER/Studio Data Archit

Modelando dados na nova versão

O ER Studio XE2 é a solução da Embarcadero para se trabalhar com arquitetura e modelagem de aplicações, processos e banco de dados. Ele caracteriza uma família de produtos, sendo composto basicamente pelas ferramentas ER/Studio Data Architect, ER/Studio Business Architect e ER/Studio Software Architect, ER/Studio Portal e ER/Studio Repository, que apesar de independentes, se integram devidamente. Fruto desta integração é o poder de combinar toda a parte de dados, processos de negócios e Design de Software num único contexto, o que irá facilitar a compreensão e a forma de projetar sistemas.

Fazendo um rápido apanhado geral sobre cada uma destas ferramentas, fica nítida a forma com que a funcionalidade de cada uma complementa a outra. Por exemplo, o ER/Studio Data Architect possui um ambiente voltado para o trabalho com modelagem de dados. Ao mesmo tempo em que as informações de metadados destes modelos poderão ser então visualizadas através de outra ferramenta, o ER/Studio Portal (ou simplesmente Portal), que é toda baseada em navegador, funcionando como uma espécie de portal colaborativo. Para a parte de modelagem de negócios e seus processos o produto envolvido é o ER/Studio Business Architect. Enquanto isso, o ER/Studio Software Architect é uma ferramenta para a modelagem de aplicativos orientados a objetos com suporte a UML. Por fim, para gerenciar tudo isso se apresenta o ER/Studio Repository (ou somente Repository), que é um aplicativo de controle de versão que funcionará de repositório principalmente para a parte de modelagem (de dados e de negócio).

A fim de evitar possíveis confusões, vale ressaltar já de início que o ER/Studio em si sofreu algumas mudanças em sua nomenclatura com o passar do tempo. Isto porque, anteriormente, o termo ER/Studio era designado à ferramenta ER/Studio Data Architect, sendo que na versão XE2 ela representa uma família de produtos por completo. Da mesma forma, o que hoje é conhecido como ER/Studio Business Architect, no passado era nomeado EA/Studio.

Resumo DevMan

De que trata o artigo:

O artigo fará um overview sobre a ferramenta voltada à modelagem de dados da Embarcadero, ER/Studio Data Architect. Da mesma forma, frisa que ela é apenas um dos integrantes do ER/Studio XE2, que é uma solução contendo outras ferramentas voltadas para áreas como modelagem de Softwares e processos de negócios. Tudo isso, com a intenção única de poder prover ao desenvolvedor opções de trabalho para áreas complementares à programação em si. O benefício neste caso em específico fica por conta do fato do produto apresentado ser da mesma fabricante da IDE de desenvolvimento utilizada no dia-a-dia, o Delphi, o que pode facilitar na busca por materiais oficiais ou mesmo em sua aquisição.

Em que situação o tema é útil:

O ER/Studio Data Architect torna-se útil a todo profissional envolvido com desenvolvimento de Software que, em sua rotina de trabalho, tenha que manipular bases de dados. A fim de se manter um controle sobre mudanças estruturais e de negócio, o profissional poderá então tirar proveito do ambiente proporcionado pela ferramenta, que é toda baseada em modelos (lógicos e físicos). Por consequência, é estabelecida uma centralização e reaproveitamento de ativos de dados, o que favorece ao desaparecimento de possíveis redundâncias. Em outros casos, consegue-se ainda manter um único modelo lógico que servirá como base para a geração de modelos físicos de bancos de dados diversos.

ER/Studio Data Architect:

Diante da realidade no cenário atual da área de desenvolvimento de Software, é possível se afirmar que, especificamente o profissional Delphi, não lida tão somente com o seu IDE de programação em seu cotidiano. Comumente ele se vê também direcionado a outros afazeres tais como levantamento de requisitos, atendimento ao cliente e gerenciamento do banco de dados. Sendo assim, almejando evitar possíveis transtornos e, acima de tudo, tentando agilizar ao máximo tais trabalhos, é interessante tomar conhecimento de técnicas e/ou ferramentas que possam vir a contribuir de alguma forma. Em se tratando especificamente da área banco de dados, vale ressaltar que a mesma fabricante do Delphi possui uma gama extensa de produtos específicos no assunto. Dessa forma, o presente artigo tem o objetivo de fazer a apresentação de um destes, o ER/Studio Data Architect, mostrando um pouco de suas principais características e mencionando sempre sua empregabilidade em exemplos de situações reais.

Modelagem de dados

A modelagem de dados pode ser considerada a primeira etapa em projetos de Softwares que tenham envolvimento com banco de dados. Isto porque, é através dela que se obtém um modelo consistente e que exponha as entidades envolvidas, bem como seus relacionamentos, que juntos irão representar as

ect XE 2
da ferramenta da Embarcadero

Fabrício Hissao Kawata

especificações do negócio propriamente dito. Além disso, uma modelagem de dados bem delineada ajuda na organização e no entendimento das informações (dados), o que acaba por reduzir a complexidade de determinado projeto. Por se tratar de uma parte importante do escopo de um sistema de informação, a modelagem de dados é subdividida em três modelos: modelo conceitual, modelo lógico e modelo físico. Cada qual se distingue mediante a perspectiva que visa atender, indo do nível mais abstrato ao mais específico.

Modelo Conceitual

O modelo conceitual pode ser considerado o estágio inicial da modelagem de dados, pois abrange o mais alto nível de abstração no que diz respeito às estruturas que serão criadas visando armazenar dados. Isto quer dizer que não há indicação do emprego de uma tecnologia específica, termos técnicos e muito menos menção ao banco de dados ou SGBD em si. Em suma, evita-se qualquer tipo de limitação e particularidade. Ao invés disso, neste momento somente serão identificadas essencialmente as prováveis entidades e a forma com que irão se relacionar mutuamente. Até mesmo por todos estes fatores, um dos objetivos principais da modelagem conceitual é a de facilitar a compreensão do modelo que está sendo elaborado, aproximando o envolvimento do cliente (usuário final). Dessa forma, ele será capaz de contribuir na definição do escopo do negócio, bem como se sentirá mais atraído pelo projeto na medida em que apresenta uma participação mais ativa.

Modelo Lógico

A partir do modelo conceitual, dá-se origem ao modelo lógico. Isto significa dizer que tudo que foi levantado no modelo primário, deverá ser levado em conta nesta segunda etapa. A diferença principal ficará por conta das limitações e recursos técnicos que irão começar a serem aplicados. Limitações estas que ainda não tem relação com qualquer plataforma de banco de dados específica. Como exemplo, é possível citar a adequação da nomenclatura de cada entidade, a definição dos campos chaves (primária e estrangeira) de cada uma delas, a criação inicial de seus atributos (campos) já com um tipo de dado e um tamanho previamente suposto e o estabelecimento de um padrão de organização. Por tudo isso, o objetivo neste ponto é fazer um refinamento, voltado para o lado técnico da representação do negócio que está sendo projetado. Sendo assim, o modelo resultante não será necessariamente o banco de dados em si, mas que será a base concreta para a construção do modelo final (físico).

Modelo Físico

No modelo físico efetivamente se dá a construção da modelagem do banco de dados que será criada, com base nas definições impostas no modelo lógico. No entanto, todos os níveis de abstração anteriores não são levados em conta neste momento, fazendo prevalecer as prováveis limitações impostas pelo SGBD a ser trabalhado. As entidades e seus atributos pressupostos até então, agora darão origem as tabelas e suas colunas, que armazenarão fisicamente os dados internamente no banco de dados. Da mesma forma, os relacionamentos impostos anteriormente, agora ganharão forma e importância, já que determinarão exatamente como os dados entre as tabelas irão se relacionar durante a execução do sistema. Além disso, é no modelo físico que aspectos como performance, agilidade, otimização devem ser levados em conta, pois a desatenção com qualquer um destes itens implicará diretamente na eficiência do banco de dados resultante. Citando como exemplo o modelo físico de uma base de dados Firebird, nele poderiam conter definições específicas de Charset e Collate para os campos descritivos que necessitem de uma particularidade maior nos casos de acentuação. Ou ainda, a criação de índices, crescentes ou decrescentes, para campos como DATA_CADASTRO e NOME, numa hipotética tabela de Clientes, já vislumbrando uma otimização na ordenação dos registros em consultas e relatórios do sistema.

ER/Studio Data Architect

O ER Studio Data Architect, ou simplesmente ER/Studio DA, é uma ferramenta destinada à modelagem visual de dados e voltada para profissionais diversos, tais como administradores de banco de dados (DBA's) e desenvolvedores. Sua IDE provê o devido gerenciamento do ciclo de vida de projetos de dados (banco de dados e metadados) de uma corporação, eliminando redundância de dados, documentando os ativos e contribuindo para as tomadas de decisões. Através dele, pode se estabelecer um único Design lógico de uma arquitetura de dados, que poderá dar origem então a N projetos físicos de bancos de dados diferentes. Elucidando uma situação real, suponha que uma tradicional Software House ofereça a seus clientes duas versões diferentes de seu sistema comercial: uma operando com o SGBD Firebird, voltado para pequenos estabelecimentos, e outra com Oracle, para empresas com fluxos maiores. Internamente ela poderia então manter um único e centralizado modelo lógico de dados, que seria a base para o modelo físico do banco de dados de cada plataforma. Num futuro, caso fosse oferecida uma terceira versão

do sistema, operando agora com MySQL, bastaria gerar um novo modelo físico para esta plataforma a partir do mesmo modelo lógico central e assim sucessivamente. Além deste, o ER/Studio DA oferece ainda alguns outros recursos, tais como:

- Sincronização de informações entre os modelos;
- Engenharia reversa de banco de dados;
- Importação e exportação de metadados;
- Geração de código DDL (Data Definition Language);
- Geração de relatórios e documentação em formatos universais, tais como HTML, XML e RTF;

Sua lista de plataformas de bancos de dados suportadas é extensa, principalmente no que diz respeito às versões de cada SGBD. Sendo assim, nominalmente falando, alguns dos principais são: Interbase, Microsoft SQL Server, MySQL, Oracle, PostgreSQL, IBM DB2, Informix Online, Microsoft Visual Fox Pro, Sybase Adaptive Server Anywhere, Sybase SQL Anywhere e Microsoft Access. Nativamente o Firebird não é listado, no entanto, assim como anteriormente no Delphi, quando se utilizava o driver Interbase em conexões dbExpress, aqui utiliza-se o suporte ao Interbase.

O ambiente

O ambiente do ER/Studio Data Architect oferece basicamente um Design orientado a modelos de dados, iniciada por padrão pela parte lógica que, posteriormente junto da parte física, provê um ideal suporte ao ciclo de vida completo de um banco de dados. A **Figura 1** mostra sua IDE, cuja interface de usuário segue o padrão Windows "look and feel", sendo formada basicamente por janelas, guias e barras de tarefas. A janela que se vê em relevo, denominada Overview é de fundamental importância em ferramentas deste tipo. Isto porque, em projetos de grande porte, com uma grande quantidade de tabelas ou entidades envolvidas, os modelos tendem a ficarem espaçosos, o que dificulta sua visualização, utilizando-se somente a barra de rolagem. Com o Overview, uma versão minimizada do modelo é exibida dentro de sua janela, tornando possível percorrer toda sua área por completo.

Gerenciando modelos

Como dito, a essência principal do ER/Studio DA é a de trabalhar com modelos de dados. Para isso, sua IDE conta com um eficiente gerenciador Treeview denominado Data Model. Ela que é apenas a primeira das várias guias dispostas na janela Data Model Explorer, localizada na parte esquerda da IDE. É através da Data Model que poderão ser manipulados os modelos (lógico e físico) que um projeto venha a possuir, o que significa criar, editar, renomear ou excluir seus objetos envolvidos. Todas estas ações são feitas através dos menus de atalhos de cada grupo de elementos. Só para citar, para a parte lógica temos como exemplo de elementos as Entidades, Atributos e Chaves, enquanto que para a parte física, as Tabelas, colunas, Triggers, Índices e quaisquer outros objetos do banco de dados em questão. A **Figura 2** mostra a Data Model carregada com a estrutura de um projeto de exemplo. Neste momento, é notória sua semelhança com a Data Explorer do próprio Delphi, o que tende a facilitar seu uso. Portanto, seja em um novo projeto iniciado no ER/Studio DA, ou mesmo numa eventual engenharia reversa de um banco de dados corrente, a Data Model será o ponto de partida com os trabalhos sobre os modelos de dados gerados.

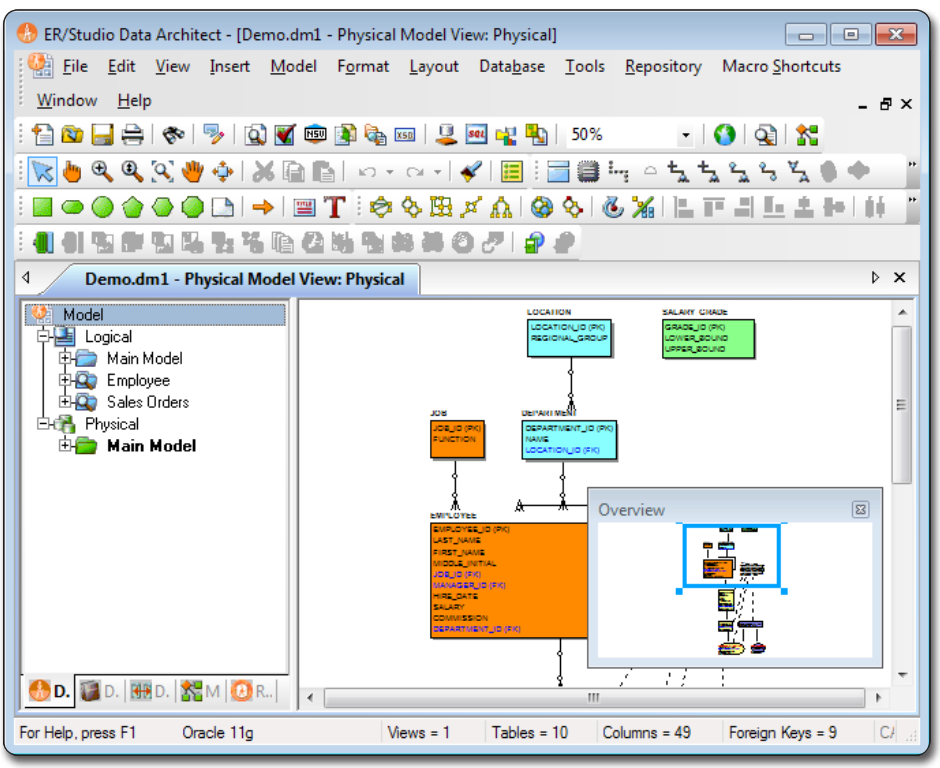


Figura 1. IDE do ER/Studio DA

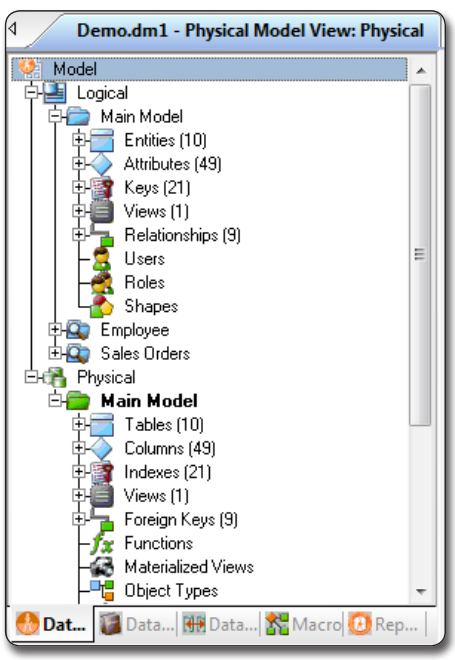


Figura 2. Data Model

Trabalhando com dicionário de dados

Um dicionário de dados (Data Dictionary) pode ser definido como sendo uma coleção de metadados contendo definições de elementos de dados. Um caso típico é a definição de Domínios (Domains) para determinados tipos de campos específicos, tais como telefone, descrição, sexo, uf, cpf, entre outros. Dessa forma, uma vez definido, este "novo" tipo de dado estabelece uma padronização, podendo ser utilizado na definição dos objetos dos modelos, como é o caso dos atributos de uma entidade e colunas de uma tabela. Na prática, um Domínio para o tipo telefone poderia então ser utilizado em todas as tabelas que possuam uma coluna referente.

Na IDE do ER/Studio DA, a definição e gerenciamento dos objetos de um dicionário de dados se dá na guia Data Dictionary, a segunda aba da janela Data Model Explorer. Nela, estão disponíveis opções para criação de anexos (Attachments), padrões (Defaults), valores de referência (Reference Values), User Datatypes, domínios (Domains) e etc. A **Figura 3** mostra um exemplo de definição de dicionário de dados no ER/Studio DA contendo, entre outras coisas, a definição dos já mencionados Domains e de um User Datatype, que nada mais é que um tipo de dado baseado em outro já existente. Neste caso, definiu-se um novo tipo booleano "genérico", baseado em um tipo Char ou Bit, a fim de evitar transtornos com possíveis SGBD's que não suportem nativamente o tipo Boolean tradicional.

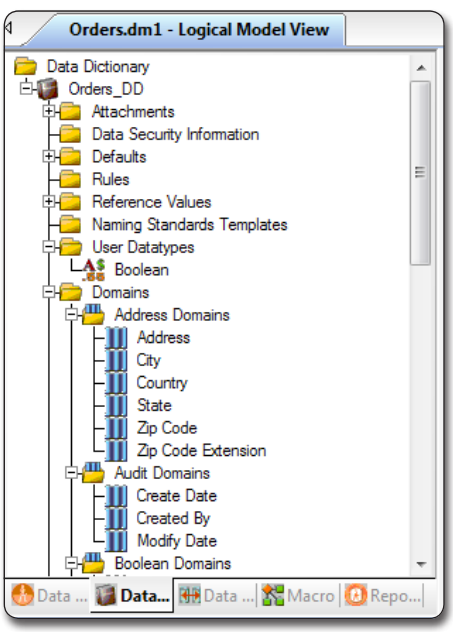


Figura 3. Data Dictionary

Documentando movimentação de dados

Data Warehouse é um termo que, em uma tradução livre para o português, quer dizer algo como Armazém ou Depósito de Dados. Em sua concepção, pode ser popularmente definido como sendo uma base de dados contendo informações extraídas de todo o ambiente de produção da empresa, em que se inclui banco de dados relacional, planilhas eletrônicas, documentos textuais e afins. Ele é utilizado para dar suporte aos processos mais críticos de tomada de decisão. Dessa forma, as informações trabalhadas reproduzem dados previamente selecionados e processados, visando única e exclusivamente a otimização de atividades de consultas, não considerando a parte transacional. Dito isto, a formação de um Data Warehouse em uma organização é algo comum e esperado, e que tem um aumento de probabilidade conforme o tamanho da empresa. Numa situação de exemplo, poderiam ser utilizadas para a formação desta estrutura múltiplas fontes de informações, provindas da Contabilidade, Folha de Pagamento, Estoque e Gestão de Relacionamento com o Cliente (CRM).

Nota do DevMan

No contexto de banco de dados, os Domains (ou domínios) podem ser definidos como uma espécie de Template (modelo) criado pelo próprio usuário, para determinados tipos de dados. Tem como objetivo estabelecer um nível ideal de padronização aos objetos de banco de dados, principalmente no que diz respeito às colunas das tabelas. Exemplos tradicionais do uso de domínios são para tipos de dados relacionados a documentos, datas, textos, telefone, etc

Para controlar todos estes aspectos do negócio e de todas as nuances relativas à como os dados se movem entre origem e destino, o ER/Studio DA fornece uma terceira guia na janela Data Model Explorer denominada Data Lineage. Através dela e de seus recursos visuais, tais como o de arrastar-e-soltar (Drag-and-Drop), é possível se documentar todo o processo de movimentação das informações, tais como suas partes de carga (Load), extração (Extraction), transformação (Transformation) e demais recursos pertinentes. É possível ainda se trabalhar com marcações de justificativas (o porquê) e de frequência de ocorrência (quando) dos processos.

Na prática, um exemplo mais trivial seria a manutenção de duas bases de dados num mesmo ambiente: uma de produção e outra exclusiva para relatórios. A primeira seria a fonte de dados para a segunda, o que significa que os dados seriam transportados de uma para outra, de forma a tirar toda e qualquer carga de processamento de relatórios da base de produção. Assim, através da Data Lineage seria possível documentar cada etapa do processo, tais como as regras específicas que determinarão a movimentação dos dados, e o tempo de vida dos mesmos, ou seja, de "quando em quando" estes dados serão arquivados. A **Figura 4** exibe a guia Data Lineage em uso.

Gerenciando Macros de execução

No jargão da computação, numa definição abrangente, as macros podem ser definidas como sendo uma série de ações canalizadas em um único procedimento que, uma vez gravadas, podem ser executadas novamente pela aplicação. Seria uma forma de fazer a aplicação lembrar exatamente das ações

executadas em um determinado momento. O ER/Studio Data Architect oferece suporte a macros de execução, que possibilita ao usuário gerenciar a acionar ações pré-determinadas que, uma vez salvas, ficam disponíveis na IDE para acesso rápido a qualquer momento. Tal gerenciamento (adição, edição e exclusão) de macros é realizado através da guia Macro, presente na Data Model Explorer.

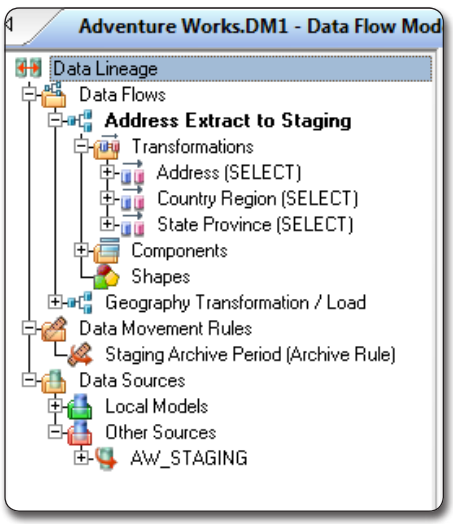


Figura 4. Data Lineage

Nativamente, o ER/Studio DA já traz algumas macros prontas, que são instaladas automaticamente com o produto e que servirão de exemplo para a definição de eventuais novas macros. Por fins de organização, elas ficam separadas em pastas, de acordo com a funcionalidade e plataforma, conforme mostra a **Figura 5**. A execução de uma macro, bem como a definição de novos scripts de automação, são realizadas no editor de macros da própria IDE, denominado ER/Studio DA Macro Editor. Para a criação de uma nova macro personalizada utiliza-se uma versão estendida de uma linguagem chamada SAX BASIC, muito semelhante ao Visual Basic. Diz-se estendida por se tratar da linguagem original com o acréscimo de tipos de objetos do próprio ER/Studio DA. A seguir é então exibido um trecho de código de uma macro padrão, destinada à exportação das informações dos Domains para uma planilha Excel.

```
Dim diag As Diagram
Dim mdl As Model
Dim dict As Dictionary
Dim dom As Domain
Dim dom_folder As DomainFolder
Dim dictionary_list () As String
...
Sub PrintCell(value As String, row As Integer, col As Integer, rowInc As Integer, colInc As
Integer, clrFore As Variant, clrBack As Variant, szFont As Integer, bBold As Boolean)
        Excel.Cells(row, col).Value = value
        Excel.Cells(row, col).Font.Bold = bBold
        Excel.Cells(row, col).Font.Color = clrFore
        Excel.Cells(row, col).Font.Size = szFont
        curRow = curRow + rowInc
        curCol = curCol + colInc
End Sub
```

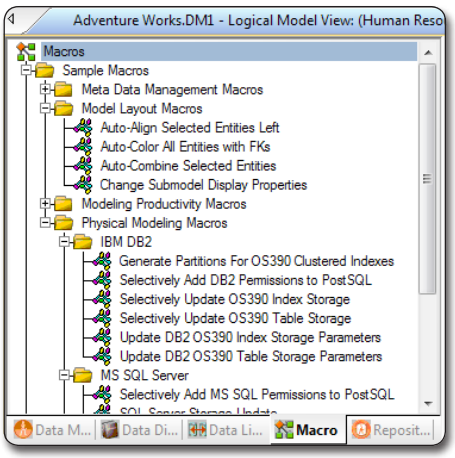


Figura 5. Guia Macro

Gerando documentação para um banco de dados

No ambiente corporativo, principalmente durante a concepção de um projeto, bem como em sua posterior vida útil, há situações onde é necessário que haja uma comunicação ideal entre os envolvidos. Dentre estes, podemos citar equipe de desenvolvimento, gerência e cliente (usuário final), que serão os responsáveis direta ou indiretamente pelo sucesso do projeto. Neste sentido, é possível afirmar que um dos pontos de sucesso diz respeito à eficiência na armazenagem das informações, que culmina na concepção do banco de dados propriamente dito. Olhando especificamente para esta parte, é importante que a documentação referente à base de dados que está sendo elaborada e mantida seja algo disponível aos interessados, seja por uma questão técnica, de conhecimento, ou qualquer outra relacionada. Sendo assim, o ER/Studio DA conta com um eficiente gerador de relatórios, acionado pela opção Generate Reports do menu Tools da IDE e capaz de prover toda esta documentação. Tal recurso trabalha essencialmente com a geração de dois tipos de relatórios, em HTML e em RTF, que são formatos que podem ser considerados mais "genéricos". O intuito disso é prover acessibilidade ao maior número de pessoas possíveis, sendo que ambos são suportados pela maioria das plataformas de Sistemas Operacionais utilizados no mercado. Ao optar pela opção HTML, ao final do processo é gerado de forma instantânea um website HTML completo, com navegabilidade funcional e acessível por qualquer navegador de internet (ex: Internet Explorer, Google Chrome, Firefox, etc.). Em contrapartida, selecionando RTF, é criado um documento individual (.rtf), que poderá ser visualizado em processadores de texto como o Microsoft Word.

Visto na prática, percebe-se que o gerador de relatórios do ER/Studio DA é na verdade um assistente (Wizard), que guia o usuário através de quatro etapas essenciais até a efetiva criação da documentação. Durante esta fase algumas configurações básicas serão determinadas, tais como o tipo de relatório desejado, o local onde o mesmo será salvo, os objetos dos modelos que farão parte da documentação, inclusão de informações do projeto, etc.

A **Figura 6** mostra um website HTML gerado a partir do gerador do ER/Studio DA, exibindo informações da tabela Customer da modelagem física de um banco de dados de demonstração. Nesta página, é possível perceber que a ferramenta já montou de forma automática toda sua estrutura, inclusive a parte de navegação Treeview (à esquerda) e estilização do layout.

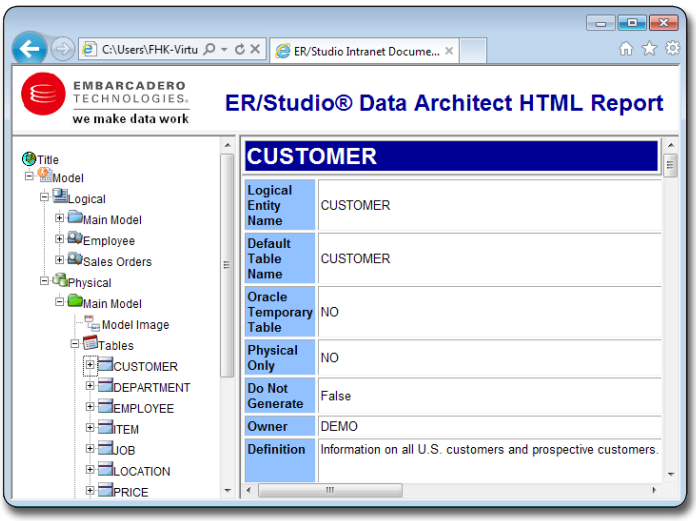


Figura 6. Relatório HTML gerado pela ferramenta

Criando modelos de dados a partir de um banco de dados existente

Pode-se dizer que o caminho natural para se trabalhar com o ER/Studio DA é a concepção de um modelo lógico consistente que servirá de base para os futuros modelos físicos de bancos de dados dos SGBD's que estarão envolvidos no negócio. No entanto, supondo o uso da ferramenta num projeto já em andamento, seria inviável ou mesmo inapropriado seguir o caminho natural citado. Justamente para casos como este é que o ER/Studio Data Architect provê o recurso de engenharia reversa, que possibilita a criação de um modelo lógico e físico a partir das informações extraídas de uma fonte de dados existente. Neste caso, é possível se especificar não somente uma conexão com uma base de dados das plataformas suportadas nativamente, como também qualquer outra, através do uso de ODBC. O que não pode deixar de ser mencionado neste ponto é que há uma diferença importante entre a escolha do tipo de conexão. Isto porque, para conexões ODBC, o ER/Studio DA só permite a engenharia reversa de tabelas e Views, enquanto que uma conexão nativa permite se manipular todos os objetos do banco. Ainda sobre as conexões nativas, a lista de plataformas que a ferramenta prevê suporte para este recurso específico é um tanto quanto restrita, envolvendo basicamente cinco tipos:

- IBM DB2 OS/390;
- IBM DB2 UDB;
- MS SQL Server;
- Oracle;
- Sybase ASE;

Dito isto, o acesso à engenharia reversa do ER/Studio DA se dá pela criação de um novo modelo básico. Contudo, ao fazer isso, a IDE já provê a opção para o aproveitamento de um banco de dados existente, conforme mostra a **Figura 7**. Ao selecioná-la, o que pode ser visto novamente é a presença de um novo assistente, denominado Reverse Enginner Wizard, que guiará o usuário por todo o restante do processo. Mais uma vez a ferramenta dá mostras de toda sua intenção em facilitar ao máximo o trabalho de seus utilizadores.

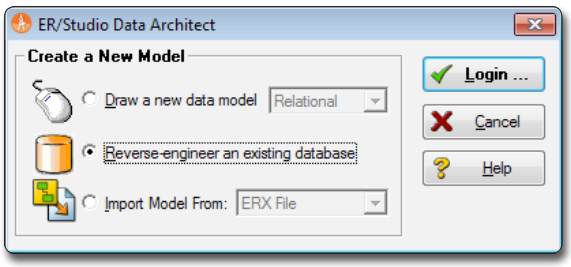


Figura 7. Opção para engenharia reversa

Portanto, é durante os passos do Wizard, que serão informados o tipo de conexão, a fonte de dados (Data Source) a ser utilizada, os objetos que deverão fazer parte do processo. Fica nítido então que, dependendo da complexidade da fonte de dados utilizada, configurar tudo isto pode se tornar trabalhoso e ainda se agravar, caso o mesmo processo (ou semelhante) necessite ser realizado com frequência. Em razão disso, na última etapa do assistente é possível salvar todas as configurações realizadas em arquivo (Settings File), para que o mesmo possa ser utilizado em processos futuros. Sem deixar de mencionar, isso é feito na primeira etapa do Wizard, onde há a opção para carregar um arquivo de definição.

Gerando modelo físico a partir de um modelo lógico

Uma vez que se constrói um modelo lógico de dados bem consistente, almeja-se imediatamente a definição dos modelos físicos de banco de dados, referentes ao SGBD's que irão ser trabalhados no projeto. Sendo assim, através do ER/Studio DA é possível realizar todo este processo de forma automática através de um gerador de modelo físico, evitando qualquer tipo de retrabalho. Isso significa que num primeiro momento toda a concentração poderá ser dada integralmente à parte lógica, visto que é garantido que ela poderá ser "transformada" integralmente em um modelo físico a qualquer momento. Este recurso provido pela ferramenta denomina-se Generate Physical Model Wizard e é acessado a partir do menu de contexto de um modelo lógico qualquer. A **Figura 8** ilustra sua interface gráfica contendo já o primeiro passo do assistente. Nesta primeira etapa fica nítida uma das principais diferenças entre modelos lógico e físico, que se traduz na indicação explícita de uma plataforma de banco de dados destino (opção "What is the target database plataform?"). Ou seja, se durante a construção do modelo lógico tem-se em mente que nenhuma particularidade ou nuance de um determinado SGBD deve ser levada em conta,

privilegiando assim uma estrutura mais genérica e abrangente, neste momento da construção do modelo físico, prima-se pelas particularidades de uma determinada plataforma.

O SGBD de destino poderá ser então qualquer uma das plataformas suportada pela ferramenta, que abre um leque de opções com mais de vinte itens. Novamente olhando para a realidade do desenvolvedor Delphi, e que tem no Firebird sua opção principal de trabalho, como dito, a ferramenta não prevê suporte nativo a esta plataforma. Dessa forma, a saída nestes casos é utilizar como plataforma de destino a opção para Interbase, da própria Embarcadero. Ao final do processo guiado pelo assistente, é criado então um novo modelo físico baseado no modelo lógico utilizado.

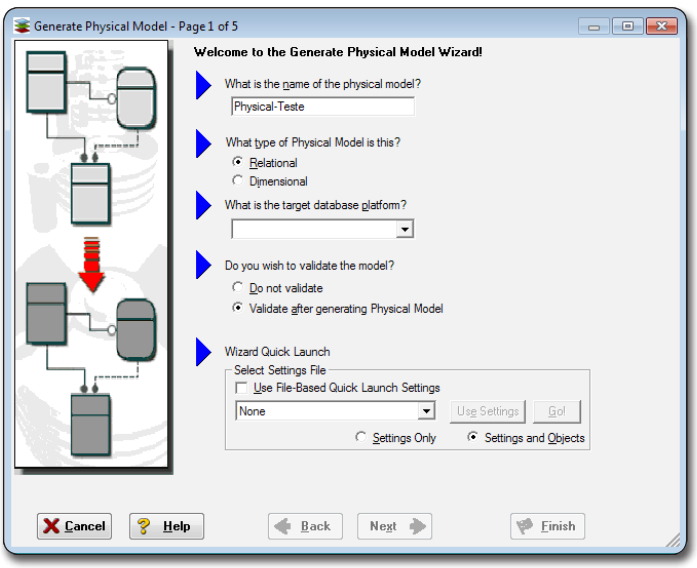


Figura 8. Assistente para geração de modelo físico baseado no modelo lógico

Atualizando modelo físico a partir do modelo lógico

Partindo do princípio de que o modelo lógico construído deverá ser a base para todos os demais modelos físicos de determinado projeto no ER/Studio Data Architect, todas eventuais alterações estruturais de dados e de negócio incidirão primeiramente na parte lógica. Tendo em vista que os modelos físicos tiveram como origem a mesma fonte, uma eventual alteração no modelo lógico deverá ser devidamente replicada em todos os modelos físicos provenientes. Isso faz com que uma harmonia seja estabelecida no projeto e, novamente, evitando possíveis retrabalhos. No ER/Studio DA, esse tipo de recurso é provido através de um utilitário assistente nomeado como "Compare and Merge Utility". Dentre suas diversas funcionalidades, ele permite comparar modelos de dados e, nos casos onde houver diferenças, estabelecer um merge entre eles. Em outras palavras, é realizada uma mescla dos pontos divergentes. Portanto, na prática, este tipo de procedimento é o que deverá ser feito para se manter modelos físicos alinhados ao modelo lógico central. A **Figura 9** exibe uma comparação realizada entre dois modelos, um lógico e outro físico. Em destaque, a indicação da diferença detectada no processo de "Compare", e sua marcação para realização de Merge.

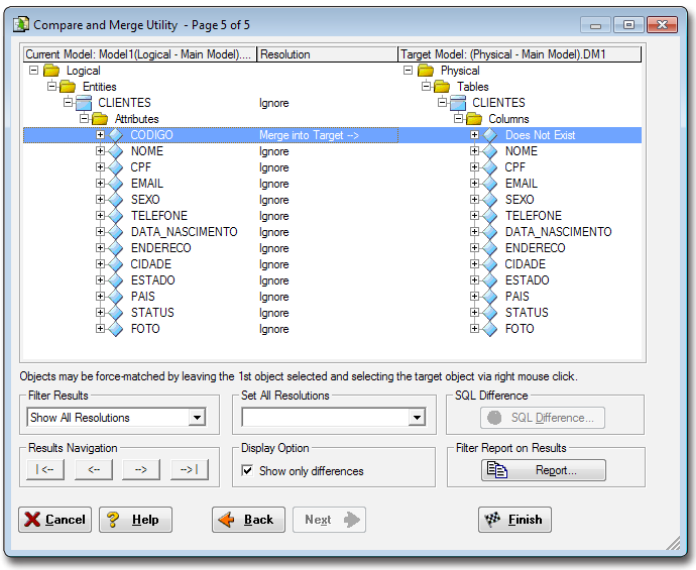


Figura 9. Compare and Merge Utility

Conclusão

O presente artigo apresentou um overview inicial de mais um dos membros da família de produtos XE2 da Embarcadero, o ER/Studio XE2. Conforme foi citado, ele não se trata de um simples produto, mas de uma solução completa no que diz respeito à modelagem de Software, dados e negócios, sendo composto de vários produtos independentes capazes de interagirem de forma agregada. Dentre estes, neste primeiro momento foi abordada a ferramenta ER/Studio Data Architect, considerada líder de indústria no cenário das ferramentas de modelagem de dados, conforme comprova sua apresentação.

Ainda neste caso em específico, olhando pelo lado prático, mesmo em projetos de longa data (já em andamento) o uso do ER/Studio DA torna-se cabível, uma vez que o mesmo oferece suporte à engenharia reversa, podendo fazer a leitura do banco de dados para modelos. Finalizando, vale ressaltar que a intenção principal novamente foi a de apresentar à comunidade Delphi ferramentas que possam vir a contribuir no dia-a-dia de desenvolvimento de Softwares. Nos próximos artigos serão feitas abordagens sobre as demais ferramentas, demonstrando a importância e contribuição de cada uma.

Fabrício Hissao Kawata
fabricio.kawata@bol.com.br
Formado em Processamento de Dados pela FATEC-TQ. Atua como Analista Programador Delphi há 6 anos. Atualmente escreve artigos sobre Delphi para revistas especializadas.
Blog: prismsilver.blogspot.com

Dê seu feedback sobre esta edição!

A Clubedelphi tem que ser feita ao seu gosto. Para isso, precisamos saber o que você, leitor, acha da revista!

Dê seu voto sobre este artigo, através do link:

www.devmedia.com.br/clubedelphi/feedback

FÓRUM DEVMEDIA

O lugar perfeito para você ficar por dentro de tudo o que acontece nas tecnologias do mercado atual

No novo Fórum da DevMedia você vai encontrar canais específicos de Delphi, ASP.NET, Java, Banco de Dados e Engenharia de Software; além de ter contato com profissionais qualificados da área para troca de informações, sugestões e muito mais.

ACESSE AGORA
www.devmedia.com.br/forum

DevMedia

FireMonkey e NFe 2.0

Uma solução passo a passo – Final

Nos artigos anteriores, deste mini-curso , foram abordados todos os passos necessários para a elaboração, assinatura, validação, envio, consulta e autorização da NF-e, o que torna o processo geral de emissão da nota fiscal eletrônica em modo normal praticamente concluído, restando apenas a geração e impressão do DANFE, o Documento Auxiliar da Nota Fiscal Eletrônica, que pode ser efetuado em papel comum contendo uma única via. Em suma, ele pode ser definido como a representação gráfica da NF-e e tem por objetivo o acompanhamento da mercadoria, bem como facilitar que o detentor ou destinatário deste documento possa consultar a NF-e perante a SEFAZ, através da chave de acesso, que também deve estar presente em forma de código de barras conforme o padrão CODE-128C, cujo o intuito é agilizar tal consulta. Entretanto, o DANFE não é uma nota fiscal e tampouco pode ser utilizado para substituí-la. Ao ser impresso em ambiente de homologação, ele deve obrigatoriamente apresentar a mensagem "SEM VALOR FISCAL" em marca d'água ou constar em Informações complementares.

Com relação às informações que devem estar presentes no DANFE, são as mesmas que compõem o documento XML da NF-e, onde não é permitida a impressão de quaisquer dados que não estiverem presentes no arquivo. Quanto à disposição delas no documento auxiliar, existem algumas regras existentes no Manual de Integração do Contribuinte que devem ser adotadas, bem como um modelo pré-definido que deve ser seguido. Também é importante frisar que este pode ser impresso tanto em modo retrato quanto paisagem, utilizando um papel cujo tamanho seja A4 no mínimo ou Ofício II no máximo. Após a emissão do DANFE, também convém que a aplicação se encarregue de enviar o e-mail para o destinatário da nota, encerrando assim todas as exigências deste processo.

Outros pontos que devem ser previstos e considerados ao elaborar a aplicação de NF-e são as adversidades, que justificam a existência de outros WebServices e implicam que o Software deve estar preparado para lidar com as mesmas, ou seja, podem haver situações onde a nota emitida necessite ser cancelada, por exemplo, lembrando também que podem ocorrer casos em que ela tenha que ser emitida em modo de contingência.

Cancelamento de NF-e

O cancelamento da nota fiscal eletrônica é um processo adicional importante que pode ser efetuado se a NF-e estiver autorizada e só é permitido em casos onde

Resumo DevMan

De que trata o artigo:

Este mini-curso tem por objetivo geral demonstrar uma forma de utilizar o FireMonkey, a nova plataforma para elaboração de aplicações presente no Delphi XE 2, para criar uma simples aplicação de NF-e 2.0 passo a passo com suporte a elaboração, assinatura, validação, envio e consulta de notas fiscais. Este último artigo tem seu foco voltado para os processos pós-envio/autorização da NF-e, demonstrando alternativas para o envio de e-mail ao destinatário da nota e impressão do DANFE, o documento auxiliar da NF-e que deve acompanhar o transporte dos produtos, bem como o cancelamento da nota se houver a necessidade.

Em que situação o tema é útil:

Utilizar os recursos do FireMonkey e do Delphi XE 2 para possibilitar o consumo de WebServices, os serviços disponibilizados através da Internet, assim como apresentar uma forma de enviar e-mails diretamente ao destinatário da nota à partir da aplicação permitindo gerar/imprimir relatórios utilizando recursos da suíte FastReports 4.0 da VCL adaptados para a aplicação no FireMonkey, possibilitando assim a impressão do DANFE apresentado neste artigo.

FireMonkey e NFe 2.0:

Com o surgimento do FireMonkey, os desenvolvedores puderam contar com novas possibilidades na elaboração de seus projetos, enriquecendo os mesmos de diversas formas. Entretanto, pelo fato de ainda ser muito recente e não possuir tantas ferramentas como a VCL, muitas vezes o desenvolvedor que adota esta plataforma, necessita implementar manualmente seus próprios códigos ou utilizar de outros recursos para ter a possibilidade de oferecer aplicações comerciais que tenham suporte a determinados recursos hoje fundamentais, fazendo com que as mesmas obtenham maior aceitação e sucesso diante do cenário atual do mercado, como é o caso de oferecer suporte a NF-e. Nesta quarta e última parte deste mini-curso são comentados os WebServices disponibilizados pela SEFAZ para oferecer suporte a outras necessidades que possam existir com o uso da NF-e. Assim como é apresentada uma forma de adaptar e utilizar os recursos da suíte FastReports 4.0, uma ferramenta completa para a elaboração de relatórios da plataforma VCL, em uma aplicação FireMonkey, onde neste exemplo é utilizado para a criação do DANFE. Para concluir, também é demonstrada uma forma de enviar e-mails através do Delphi XE 2, sendo este utilizado para o envio do documento XML da NF-e para o destinatário.

a mercadoria ainda não foi enviada, lembrando que a SEFAZ libera o cancelamento da nota pelo período de 24 horas a partir da autorização da mesma. Este processo também tem seu funcionamento baseado em um WebService específico denominado NfeCancelamento2, sendo o mesmo do tipo síncrono, o que implica que este processo é concluído em uma única solicitação. Para efetuar o cancelamento de uma nota, a aplicação deve enviar uma mensagem baseada no padrão SOAP de acordo com o XML Schema denominado "cancNFe_v2.00.xsd", contendo o pedido de cancelamento, que por sua vez deve possuir a chave de acesso, o número de protocolo de autorização da NF-e, uma justificativa, dentre outras informações, lembrando que para a efetivação deste processo, o documento XML do pedido também deve ser assinado com o certificado digital do emitente para comprovar a validade do mesmo. Outro ponto que deve ser levado em consideração pelo desenvolvedor é que este processo deve ser realizado nota a nota, não permitindo que seja enviado um lote contendo diversas. Entretanto, se houver a necessidade de corrigir erros ou adicionar alguma informação em uma nota, não implica que a mesma deva ser cancelada. Existem alternativas para este fator, como é o caso da emissão de Nota Fiscal Eletrônica Complementar ou a Carta de Correção Eletrônica, que podem ser adotadas.

Para dar sequência à implementação deste passo, pode-se abrir o mesmo projeto que vem sendo utilizado para este mini-curso. Com ele aberto, deve-se efetuar o XML Data Binding do Schema correspondente ao mesmo citado anteriormente (cancNFe_v2.00.xsd) para possibilitar uma melhor manipulação do documento XML que será enviado. O recurso de XML Data Binding pode ser facilmente acessado através do menu File>New>Other>XML>XML Data Binding. Este recurso permite a criação automática das interfaces, seus respectivos atributos e métodos com base no documento XML, gerando assim a Unit cancNFe_v200.pas. A **Listagem 1** apresenta um procedimento para o cancelamento de uma NF-e, recebendo os dados necessários para o mesmo, sendo eles a chave de acesso, o número do protocolo e a justificativa do cancelamento.

Neste procedimento, após a declaração das variáveis entre as linhas 2 e a 12, o objeto XMLDocCancelamento do tipo TXMLDocument é devidamente inicializado (linha 14), lembrando que este tipo de objeto é utilizado para a manipulação de documentos XML. Após a ativação do objeto anterior na linha 15, o objeto CancNFe recebe a interface do elemento que é obtida através da função GetDocBinding(), presente na linha 16. Em seguida, entre as linhas 17 e 29, são definidas as propriedades do mesmo, como o Id, TpAmb, XServ, etc,

Nota do DevMan

O XML (Extensible Markup Language) é um dos padrões mais adotados para o compartilhamento de informações através da Internet, sendo constituído por uma estrutura de marcação. O XML possibilita uma separação entre a formatação e os dados e simplifica a organização das informações. Por ser baseado em texto, é facilmente utilizado em diversas linguagens de programação e pode ser compartilhado entre vários ambientes operacionais como Windows, Linux, entre outros.

Nota do DevMan

O CODE-128 é um padrão simbólico linear adotado para ser empregado na impressão de códigos de barras alfanuméricos, permitindo a codificação dos 128 caracteres ASCII. O subtipo C é utilizado para a simbologia numérica com dígito verificador. Este é o padrão adotado na impressão do DANFE.

lembrando que este objeto (CancNFe) é utilizado para montar a mensagem de cancelamento que será enviada. A variável ArquivoXml recebe a string XML do objeto XMLDocCancelamento na linha 30, através da propriedade XML.Text. Terminado este processo, os procedimentos SelecionarCertificado (linha 32) e AssinarNFe (linha 33), que foram apresentados no Artigo 2 deste mini-curso, podem ser reaproveitados facilmente. A seguir, na linha 35, é utilizado um arquivo de texto (TextFile) para carregar o documento XML que foi gerado pelo procedimento AssinarNFe, uma vez que o mesmo foi gravado no disco, ele é lido através de Readln na linha 37 e tem seu conteúdo armazenado na variável ArquivoXml, sendo posteriormente fechado na linha 38 através de CloseFile. Na linha 39, o objeto HTTPReqResp recebe uma instância da classe THTTPReqResp (Create) e algumas de suas propriedades são definidas entre as linhas 40 e 46, tais como opções adicionais através de InvokeOptions, URL e ação SOAP, responsável por efetuar a chamada de método do WebService. O THTTPReqResp é um componente que pode ser utilizado para o consumo de um WebService, possibilitando o envio da requisição e consequentemente o retorno da mesma, utilizando uma mensagem no padrão SOAP. Entre as linhas 47 e 62 a mensagem SOAP (Msg SOAP) que será enviada para o Webservice é definida, recebendo além das informações padrões necessárias, a String do XML que precisa ser enviada. Posteriormente, na linha 64 é criada a variável Lista do tipo TStringList, que armazenará a mensagem SOAP. Tal mensagem é atribuída a propriedade Text da variável Lista na linha 65. Em seguida, na linha 66 o objeto MemoriaStream recebe uma instância da classe TMemoryStream (Create) com o intuito de possibi-

litar a utilização do método Execute do HTTPReqResp (linha 67) para iniciar o consumo do WebService. O TMemoryStream é importante para possibilitar que informações possam ser manipuladas em memória, permitindo também que a mesma seja exportada. Na linha 68, o objeto TextoStream recebe uma instância da classe TStringStream (Create) e copia o conteúdo da variável MemoriaStream através do método CopyFrom(), presente na linha 69, oferecendo recursos para a manipulação de Strings. Na linha 70, a variável Resultado recebe o conteúdo String do objeto TextoStream, através da propriedade DataString. Em seguida, o arquivo XML de retorno é gravado em um diretório do disco através do método SaveToFile(), presente na linha 71, posteriormente exibindo uma mensagem (linha 72) contendo a informação da TAG xMotivo, que por sua vez contém uma das principais informações que são retornadas pelo WebService. Concluindo, , entre as linhas 73 e 78 os objetos são liberados da memória (FreeAndNil).

Inutilização de NF-e

O processo de Inutilização de NF-e é uma alternativa que pode ser aplicada sempre que ocorrerem problemas na aplicação do emitente ou de ordem técnica, que possam ocasionar uma quebra na sequência do número da NF-e. Sendo assim, a faixa de números que não será utilizada entre a última nota emitida e a nova que será elaborada, deve ser informada à SEFAZ até o décimo dia do mês seguinte. Vale à pena evidenciar também que as notas que forem canceladas não devem ter seu número inutilizado. Este processo pode ser efetuado através do envio do documento XML, utilizando uma mensagem SOAP para o WebService NfeInutilizacao2. Este processo também pode ser implementado facilmente de maneira semelhante ao CancelarNFe apresentado anteriormente, evidentemente alterando alguns fatores como o conteúdo do documento XML, o WebService, a ação SOAP, etc.

Listagem 1. Procedimento para o envio do documento XML da NF-e através de E-mail

```
01. procedure CancelarNFe(ChaveNFe, Protocolo, Justificativa: string);
02. var
03.   MemoriaStream: TMemoryStream;
04.   TextoStream: TStringStream;
05.   MsgSOAP, Resultado: AnsiString;
06.   Lista: TStringList;
07.   HTTPReqResp: THTTPReqResp;
08.   PosId: Integer;
09.   CancNFe: IXMLTCancNFe;
10.  XMLDocCancelamento: TXMLDocument;
11.  ArquivoXml: AnsiString;
12.  Arquivo: TextFile;
13.  begin
14.  XMLDocCancelamento := TXMLDocument.Create(nil);
15.  XMLDocCancelamento.Active := True;
16.  CancNFe := XMLDocCancelamento.GetDocBinding('cancNFe', TXMLTCancNFe,
       'http://www.portalfiscal.inf.br/nfe') as IXMLTCancNFe;
17.  with CancNFe do
18.  begin
19.    Versao := '2.00';
20.    with InfCanc do
21.    begin
22.      Id := 'ID'+ ChaveNFe;
23.      TpAmb := '2';
24.      XServ := 'CANCELAR';
25.      ChNFe := ChaveNFe;
26.      NProt := Protocolo;
27.      XJust := Justificativa;
28.    end;
29.  end;
30.  ArquivoXml := XMLDocCancelamento.XML.Text;
31.  XMLDocCancelamento.Active := False;
32.  SelecionarCertificado;
33.  if AssinarNFe(CertificadoAtual, ArquivoXml, 'Canc') then
34.  begin
35.    AssignFile(Arquivo, 'C:\FMXNFE\NFeACancelar\'+ ChaveNFe +'-canc.xml');
36.    Reset(Arquivo);
37.    ReadIn(Arquivo, ArquivoXml);
38.    CloseFile(Arquivo);
39.    HTTPReqResp := THTTPReqResp.Create(Self);
40.    with HTTPReqResp do
41.    begin
42.      InvokeOptions := [soIgnoreInvalidCerts,soNoSOAPActionHeader,
         soPickFirstClientCertificate,soAutoCheckAccessPointViaUDDI];
43.      URL := 'https://homologacao.nfe.fazenda.sp.gov.br/nfeweb/services/
          NfeCancelamento2.asmx';
44.      UseUTF8InHeader := True;
45.      SoapAction := 'http://www.portalfiscal.inf.br/nfe/wsdl/NfeCancelamento2';
46.    end;
47.    MsgSOAP := '<?xml version="1.0" encoding="utf-8"?>'+
48.          '<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/
              XMLSchema-instance"'+
49.          'xmlns:xsd="http://www.w3.org/2001/XMLSchema"'+
50.          'xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">'+
51.          '<soap12:Header>'+
52.          '<nfeCabecMsg xmlns="http://www.portalfiscal.inf.br/nfe/wsdl/
               NfeCancelamento2">'+
53.          '<cUF>35</cUF>'+
54.          '<versaoDados>2.00</versaoDados>'+
55.          '</nfeCabecMsg>'+
56.          '</soap12:Header>'+
57.          '<soap12:Body>'+
58.          '<nfeDadosMsg xmlns="http://www.portalfiscal.inf.br/nfe/wsdl/
               NfeCancelamento2">'+
59.          ArquivoXml +
60.          '</nfeDadosMsg>'+
61.          '</soap12:Body>'+
62.          '</soap12:Envelope>';
63.    try
64.      Lista := TStringList.Create;
65.      Lista.Text := MsgSOAP;
66.      MemoriaStream := TMemoryStream.Create;
67.      HTTPReqResp.Execute(Lista.Text, MemoriaStream);
68.      TextoStream := TStringStream.Create('');
69.      TextoStream.CopyFrom(MemoriaStream, 0);
70.      Resultado := UTF8Decode(TextoStream.DataString);
71.      TextoStream.SaveToFile('C:\FMXNFE\NfeCanceladas\' + ChaveNFe + '-canc.xml');
72.      ShowMessage(getConteudoTAG(Resultado, 'xMotivo', ''));
73.    finally
74.      FreeAndNil(HTTPReqResp);
75.      FreeAndNil(MemoriaStream);
76.      FreeAndNil(TextoStream);
77.      FreeAndNil(Lista);
78.    end;
79.  end;
80. end;
```

Nota do DevMan

O WebService pode ser definido como um recurso que disponibiliza diversos serviços gratuitos ou pagos através da Internet. Ele tem seu funcionamento realizado de forma automática, onde a requisição da aplicação cliente é recebida e processada, sendo retornada uma resposta logo em seguida. É baseado nos padrões HTTP e XML, permitindo assim grande compatibilidade com diversos Sistemas Operacionais, implementando também o protocolo SOAP.

Nota do DevMan

O HTTP (HyperText Transfer Protocol) é, em suma, o protocolo padrão de transporte adotado na Internet para permitir a navegação entre sites, transferência de informações e etc. É responsável por tratar as solicitações e respostas entre o servidor e o cliente.

Nota do DevMan

O protocolo SOAP (Simple Object Access Protocol) é um padrão adotado para a transferência de informações entre o WebService e a aplicação cliente, baseado em XML (estrutura) e HTTP (transporte). Para efetuar a troca dos dados é utilizada uma mensagem de acordo com o funcionamento do WebService que será consumido, contendo informações como o método que deve ser executado, os dados necessários para a requisição, entre outras informações.

Elaboração do DANFE através do FireMonkey

Por ser uma plataforma muito recente, o FireMonkey ainda não possui oficialmente uma ferramenta de relatórios completa nativa, porém vale ressaltar que atualmente existem alguns aplicativos que prometem converter componentes VCL para FireMonkey. Entretanto, existem inúmeras maneiras de elaborar o DANFE para ser utilizado pela aplicação desenvolvida nesta plataforma, mesmo que não tenha uma ferramenta específica. Atualmente, existem muitos sites na Web que oferecem o serviço de geração do documento auxiliar gratuitamente, onde o desenvolvedor simplesmente submete o documento XML da NF-e autorizada e pode obter o documento auxiliar em vários formatos, como é o caso do WebDANFE (ver seção links), dentre outros. O desenvolvedor também pode contar com aplicações executáveis gratuitas, desenvolvidas na VCL ou em outros ambientes de programação que oferecem o recurso de geração do DANFE. Contudo, apesar de serem soluções prontas, fáceis e ágeis, tais recursos acabam descentralizando a aplicação de NF-e, uma vez que existe uma "terceirização" para a elaboração do documento auxiliar. Como o objetivo deste projeto é ser desenvolvido totalmente no ambiente FireMonkey, existem alguns métodos alternativos que possibilitam a elaboração do documento auxiliar diretamente nesta plataforma, dos quais se destacam a geração do DANFE em HTML ou principalmente PDF (Portable Document Format), que foi o padrão adotado neste exemplo. Para possibilitar a criação do arquivo no formato PDF, alguns componentes do FastReports 4.0 presentes no Delphi XE 2 podem ser utilizados. O FastReports é uma ferramenta completa para elaboração, edição, visualização e impressão de relatórios para a plataforma VCL de uma maneira simples e rápida, através de um editor visual. Apesar de ser destinado a outra plataforma, com algumas alterações é possível utilizar componentes do mesmo para o FireMonkey quando o ambiente alvo é o Windows, como é o caso dos objetos TfrxReport e TfrxPDFExport. O componente TfrxReport é o objeto principal que contém todas as ferramentas para o carregamento, visualização e manipulação dos arquivos de relatórios do FastReport. Está presente na Unit frxClass. O componente TfrxPDFExport, presente na Unit frxPDFExport, é um objeto voltado para a exportação de um relatório diretamente para um arquivo PDF, sendo neste caso muito útil para a elaboração do documento auxiliar no FireMonkey e deve ser utilizado em conjunto com o componente TfrxReport.

Como o DANFE será elaborado através do FastReport, o primeiro passo é abrir o Software Designer do mesmo e iniciar a criação de um novo relatório, que por sua vez ao ser concluído será gravado em um arquivo no formato .fr3. Para abrir o Designer, se ele não estiver presente no menu Iniciar, pode-se criar um projeto temporário VCL Forms Application apenas para abrir o editor integrado e possibilitar a criação e gravação do arquivo do relatório, inserindo posteriormente um componente TfrxReport da aba FastReports 4.0 no formulário e efetuando um clique duplo sobre o mesmo. Com o Desginer aberto, pode-se iniciar a elaboração do DANFE levando em consideração as exigências da receita presentes no Manual do Contribuinte (consulte a seção de links para maiores informações sobre edição de relatórios com o FastReports e o Manual do Contribuinte). Entretanto, existem alguns projetos que disponibilizam o DANFE no formato .fr3 pronto e gratuitamente, necessitando apenas de uma adaptação para serem utilizados em qualquer aplicação.

Para iniciar a criação de um novo relatório no FastReport, acesse o menu File>New do editor e selecione Standard Report. Posteriormente deve-se definir as margens do mesmo através do menu File>Page Settings, ajustando Left/Right para 0,50 e Top/Bottom para 0,70. Para não estender a criação do DANFE, uma vez que o mesmo possui diversos itens e regras, neste exemplo será criada apenas a área destacável do documento auxiliar que é assinada pelo receptor. Contudo, para implementá-lo completamente basta seguir os mesmos passos que serão descritos, preenchendo com as informações necessárias. Devem ser inseridos uma banda do tipo ReportTitle, através do botão InsertBand e 5 Memos para o preenchimento das informações. Para efetuar a futura transferência dos dados entre a aplicação e a ferramenta de relatório, neste exemplo serão utilizados os recursos de variáveis (Variables) e de DataSets do FastReport, com o intuito de demonstrar como efetuar a integração com ambos através do FireMonkey. Algumas das informações do Emitente e do Destinatário serão criadas futuramente no Delphi com uso de DataSet e outros dados como número de série da NF-e, data e valor total, serão criados através de variáveis. Para criar variáveis para este relatório basta acessar o menu Report>Variables do Designer. Na janela Edit Variables deve-se criar uma nova categoria através do botão Category, que neste exemplo foi renomeada para "Geral".

Nesta categoria também devem ser criadas as quatro variáveis que serão utilizadas através do botão Variable, lembrando que a renomeação de variáveis e categorias no editor do Fast Report pode ser realizada através do botão Edit ou da tecla F2. A **Figura 1** demonstra a janela Edit Variables com as quatro variáveis que foram criadas para este exemplo.

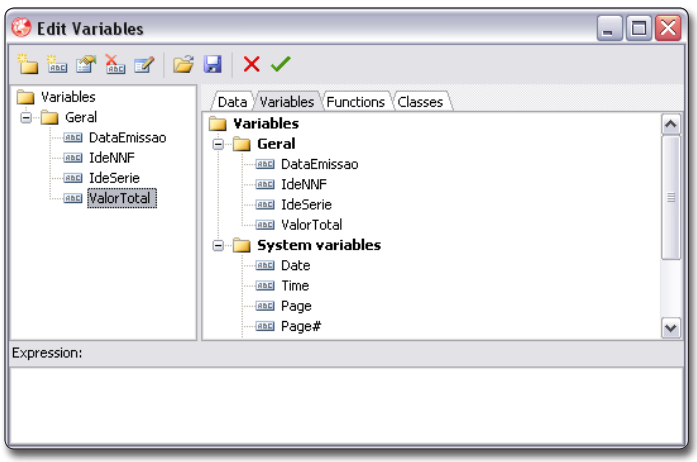


Figura 1. Janela Edit Variables da aplicação Designer e variáveis criadas

Foram criadas as variáveis DataEmissao, IdeNNF, IdeSerie e ValorTotal. Com isto o Memo1 adicionado anteriormente terá seu preenchimento efetuado através da propriedade Memo, de acordo com o código a seguir:

```
Recebemos de [frxDataSet."EmitXNome"] os produtos e/ou serviços constantes da Nota
Fiscal Eletrônica indicada ao lado.
Destinatário: [frxDataSet."DestXnome"] - [frxDataSet."DestXLgr"], [frxDataSet."DestNro"] -
[frxDataSet."DestXBairro"] - [frxDataSet."DestXMun"] - [frxDataSet."DestUF"].
Data de Emissão: [DataEmissao]  Valor Total da Nota: R$ [ValorTotal]
```

Neste exemplo os campos semelhantes a [frxDataSet."EmitXNome"] são utilizados para a obtenção das informações diretamente de um DataSet, que será futuramente criado no Delphi e receberá o nome de frxDataSet para o relatório. Já os valores dos campos que contém a descrição semelhante a [DataEmissao], serão carregados através das variáveis que foram criadas no Designer do FastReport. Esta é uma grande vantagem ao utilizar o FastReport, ou seja, é possível preencher as informações diretamente através de texto e palavras-chave.

Com relação ao Memo5, o mesmo é preenchido da seguinte forma:

```
Nº [IdeNNF]
Série [IdeSerie]
```

Este trecho permite o futuro preenchimento das variáveis [IdeNNF] e [IdeSerie] via código através do FireMonkey. Após estes passos, a **Figura 2** apresenta o visual final do relatório desta primeira etapa de elaboração do DANFE com os tamanhos dos Memos devidamente ajustados.

Para concluir esta etapa, o relatório foi gravado com o nome de "DANFE.fr3" no mesmo diretório que contém o executável deste projeto para que futuramente possa ser carregado facilmente em tempo de execução.

A próxima etapa para a geração do DANFE é inserir um novo formulário, no mesmo projeto utilizado neste mini-curso, para administrar toda a elaboração do DANFE. Deve-se criar um TClient DataSet, o objeto que permite trabalhar com as informações na memória, ou qualquer outro tipo de DataSet se o desenvolvedor optar pela conexão com o banco de dados, possibilitando assim a integração com o relatório em FastReport. O ClientDataSet foi renomeado para cdsDados e a propriedade FieldsDefs foi utilizada para a criação dos campos do mesmo de forma independente. **A Figura 3** demonstra os campos criados no cdsDados através do FieldsEditor da propriedade FieldsDef.

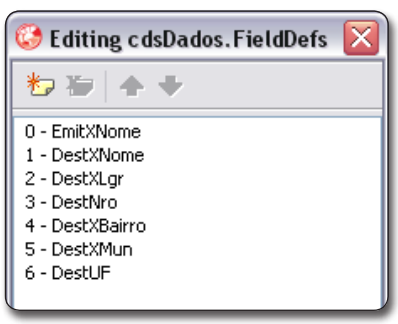


Figura 3. Campos do cdsDados

Uma vez que os campos estão criados, o próximo passo é clicar com o botão direito sob o cdsDados e selecionar Create-DataSet, para finalizar a criação dos mesmos e posteriormente, através de um clique duplo sob o cdsDados, adicionar todos os

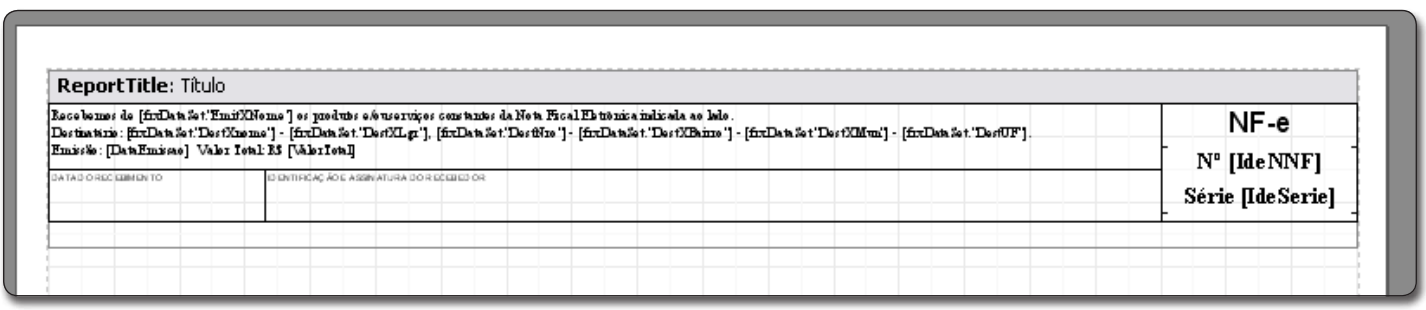


Figura 2. Visual da primeira parte do relatório de impressão do DANFE

campos criados utilizando a opção Add All Fields do menu de contexto deste objeto. Com o ClientDataSet criado, é elaborado um procedimento cuja função será carregar o documento XML da NF-e autorizada e preencher suas informações. Para tanto, será também utilizada a mesma função auxiliar apresentada no artigo 3 deste mini-curso. A **Listagem 2** demonstra a função getConteudoTAG cujo propósito é obter dados presentes em documentos XML facilmente.

Listagem 2. Função para obter dados de documentos XML

```
function TFormMovEnvNotaFiscal.getConteudoTAG(ArquivoXML, NomeTAG,
pTAG: AnsiString): AnsiString;
var
  PosIni, PospTAG: Integer;
begin
  if Trim(pTAG) <> '' then
  begin
    PospTAG := Pos('<'+ pTAG, ArquivoXML);
    ArquivoXML := Copy(ArquivoXML, PospTAG, Pos('</'+ pTAG, ArquivoXML)
    - PospTAG);
  end;
  PosIni := Pos('<'+ NomeTag, ArquivoXML) + Length(NomeTag) + 2;
  Result := Copy(ArquivoXML, PosIni, Pos('</'+ NomeTag, ArquivoXML) - PosIni);
end;
```

Esta função tem como parâmetros as variáveis ArquivoXML que recebem a String do XML, NomeTAG onde é informada a TAG que contém o valor desejado e pTAG, utilizada para a especificação de um nó "pai", uma vez que o documento pode ter várias TAGs filhas de mesmo nome em TAGs pais diferentes. Sendo assim, é efetuada a verificação da pai (pTAG), onde o preenchimento da mesma implica em uma pré-filtragem das informações através da função Pos, cujo objetivo é retornar a posição da primeira ocorrência de determinada String. Na sequência, a função Pos é reutilizada para obter a primeira ocorrência da TAG informada no parâmetro NomeTAG. Posteriormente é somado o tamanho (Length) da mesma com o valor dos delimitadores (< >). Para concluir, o resultado desta função é obtido através do Copy, tendo como parâmetros a String XML, a posição inicial e a posição da informação que é obtida através da posição da TAG de encerramento (</ >) subtraída da posição inicial.

Nota do DevMan

Uma TAG (etiqueta) pode ser definida como uma palavra-chave que é utilizada para a organização de dados presentes em linguagens de programação e arquivos, que podem ser palavras reservadas ou criadas pelo próprio desenvolvedor, como é o caso do XML. Pode ser constituída por <nome da TAG + atributos>dados</nome da TAG>.

Dando continuidade à implementação do DANFE com o Fast Report, os objetos TfrxReport e TfrxPDFExport mencionados anteriormente deverão ser utilizados. Para tanto, as Units frxClass e frxPDFExport devem ser incluídas no uses do novo formulário. Contudo, ao tentar compilar o projeto após a inclusão destas Units é apresentado um erro nas declarações das mesmas, uma vez que estão sendo utilizadas na plataforma FireMonkey. Para resolver este problema basta adicionar a palavra "Vcl." antes de cada declaração de Unit do FastReport que apresentar erro. Criamos o procedimento chamado CriarDANFE (**Listagem 3**), ele é responsável por carregar o documento XML, preencher o cdsDados, efetuar a exportação do relatório para PDF e permitir a visualização do DANFE utilizando a aplicação PDF padrão. Basicamente todo relatório FastReport está conectado a um conjunto de dados, neste exemplo essa conexão se dá através do componente TfrxDBDataset (variável FastDBDataSet). Através de seu uso é possível conectar um relatório com qualquer fonte, desde que essa fonte possa ser acessada como um TDataSet. Esse tipo de objeto pode se relacionar com qualquer descendente de TDataSet. Nesse método o nome do arquivo é passado e carregado no TClient DataSet cdsDados. Como ambos estão ligados entre si (linha 19), FastDBDataSet funciona para o relatório como uma ponte de acesso aos dados.

Neste procedimento, após as declarações das variáveis entre as linhas 2 e 8, o objeto Relatorio recebe uma instância da classe TfrxReport (Create) na linha 10, assim como na linha 11 o objeto FastPDF também recebe sua devida instância (TfrxPDFExport), o FastReport possui classes que tratam da exportação de um relatório para os mais diversos formatos. Na linha 13, a função Pos() é utilizada para encontrar a primeira ocorrência do atributo id ("id=") presente na variável ArquivoXML para possibilitar a obtenção da chave de acesso da NF-e através do comando Copy() (linha 14) com o objetivo de ser empregada futuramente para o nome do arquivo PDF que será criado. Na linha 15, o endereço do diretório atual da aplicação é armazenado na variável Diretorio com a utilização de ExtractFileDir(). Para efetuar a integração com o arquivo de relatórios criado anteriormente (DANFE.fr3), o objeto FastDBDataSet (TfrxDataSet) teve sua propriedade Name (linha 18) atribuída com o mesmo nome que foi definido no Memo1 do relatório (frxDataSet) e a propriedade DataSet (linha 19) atribuída com o objeto cdsDados. Efetuado este processo, entre as linhas 23 e 36, o cdsDados é preenchido com algumas informações do arquivo XML da NF-e com o auxílio da função getConteudoTAG, tendo os dados gravados em memória. Na sequência, o arquivo de relatório criado anteriormente é carregado diretamente do disco para o objeto Relatorio (TfrxReport) através da função LoadFromFile, presente na linha 39. O DataSet criado anteriormente (FastDBDataSet) é adicionado ao relatório para a integração com o cdsDados através do método Relatorio.DataSets.Add (linha 40). Posteriormente, entre as linhas 43 e 46, as variáveis criadas no relatório são preenchidas através da propriedade "Variables[]" do objeto Relatorio e o mesmo é preparado com o uso do método PrepareReport, conforme a linha 47. Em seguida, na linha 49, a variável NomeArquivo recebe o endereço que será utilizado para a gravação do arquivo PDF, bem como seu nome. O objeto FastPDF

(TfrxPDFExport) tem a propriedade FileName definida na linha 52, o local onde o arquivo PDF será gravado no disco, desativando a janela de diálogo "Salvar" através de ShowDialog := False (linha 53). Em seguida, o relatório é exportado para PDF na linha 55 com o uso de Relatorio.Export(FastPDF), sendo posteriormente efetuada a chamada para o visualizador de PDF padrão instalado no sistema através da função ShellExecute conforme a linha 56, lembrando que esta função permite a abertura de programas externos através da aplicação. Para concluir, entre as linhas 58 e 60, os objetos são liberados da memória.

Como observado neste processo, a integração entre o FireMonkey e um relatório elaborado no FastReport do Delphi XE 2 pode ser realizada sem maiores complicações através da utilização de alguns objetos criados em tempo de execução e da definição de suas propriedades de acordo com a necessid ade do projeto, bem como o carregamento de um arquivo de relatório criado e gravado.

Envio do documento XML através de E-mail

A etapa final do processo geral de emissão de NF-e em modo normal é o envio do e-mail com o documento XML anexado para o destinatário, uma vez que este é obrigatório por parte do emitente. O Delphi traz consigo diversos componentes do Projeto Indy, um projeto Open Source iniciado há muito tempo por voluntários e que oferece uma biblioteca de diversos objetos, disponibilizando assim a estrutura necessária para a utilização de e-mail na ferramenta, dentre outros diversos recursos. O componente que pode ser utilizado para o envio de mensagens eletrônicas é o TIdSMTP, presente na Unit IdSMTP. Este componente oferece diversas funcionalidades e configurações necessárias para a conexão com o servidor e envio de e-mails com a implementação do protocolo SMTP (Simple Mail Transfer Protocol), permitindo assim a elaboração de uma aplicação cliente. O SMTP por sua vez, é um protocolo padrão baseado em texto adotado na Internet que tem como função possibilitar a saída de mensagens eletrônicas, normalmente operando na porta 25, mas pode variar de acordo com o servidor, sendo também baseado no protocolo TCP/IP. Sendo assim, o objeto TIdSMTP deve ser configurado de acordo com o servidor que será usado. Quando existe a necessidade do uso de SSL (Secure Socket Layer) para possibilitar uma conexão segura com o servidor, um objeto TIdSSLIOHandlerSocketOpenSSL, pertencente a Unit

Listagem 3. Procedimento para a criação do DANFE

```
01. procedure TFormImpDanfe.CriarDANFE(ArquivoXML: AnsiString);
02. var
03.   PosId: Integer;
04.   ChaveNFe, Diretorio, NomeArquivo, dtEmissao: string;
05.   AuxData: TDate;
06.   Relatorio: TfrxReport;
07.   FastPDF: TfrxPDFExport;
08.   FastDBDataSet: TfrxDBDataset;
09. begin
10.  Relatorio := TfrxReport.Create(nil);
11.  FastPDF := TfrxPDFExport.Create(nil);
12.  FastDBDataSet := TfrxDBDataset.Create(nil);
13.  PosId := Pos('Id=', ArquivoXML) + 7;
14.  ChaveNFe := Copy(ArquivoXML,PosId, 44);
15.  Diretorio := ExtractFileDir(GetCurrentDir);
16.  with FastDBDataSet do
17.  begin
18.    Name := 'frxDataSet';
19.    DataSet := cdsDados;
20.  end;
21.  try
22.    //Preenchimento dos dados do DataSet
23.    with cdsDados do
24.    begin
25.      Close;
26.      CreateDataSet;
27.      Append;
28.      cdsDadosEmitXNome.AsString := getConteudoTAG(ArquivoXML,
          'xNome','emit');
29.      cdsDadosDestXNome.AsString := getConteudoTAG(ArquivoXML,
          'xNome','dest');
30.      cdsDadosDestXLgr.AsString := getConteudoTAG(ArquivoXML,'xLgr','dest');
31.      cdsDadosDestNro.AsString := getConteudoTAG(ArquivoXML,'nro','dest');
32.      cdsDadosDestXBairro.AsString := getConteudoTAG(ArquivoXML,
          'xBairro','dest');
33.      cdsDadosDestXMun.AsString   := getConteudoTAG(ArquivoXML,'xMun','dest');
34.      cdsDadosDestUF.AsString     := getConteudoTAG(ArquivoXML,'UF','dest');
35.      Post;
36.    end;
37.    with Relatorio do
38.    begin
39.      LoadFromFile('C:\FMXNFE\DANFE.fr3');
40.      DataSets.Add(FastDBDataSet);
41.      AuxData := Date;
42.      //Preenchimento das variáveis
43.      Variables['DataEmissao'] := QuotedStr(FormatDateTime('dd/mm/yyyy', AuxData));
44.      Variables['IdeNNF']    := QuotedStr(getConteudoTAG(ArquivoXML,'nNF','ide'));
45.      Variables['IdeSerie']  := QuotedStr(getConteudoTAG(ArquivoXML,'serie','ide'));
46.      Variables['ValorTotal'] := QuotedStr(getConteudoTAG(ArquivoXML,'vNF','total'));
47.      PrepareReport;
48.    end;
49.    NomeArquivo := 'C:\FMXNFE\DANFE\'+ ChaveNFe +'.pdf';
50.    with FastPDF do
51.    begin
52.      FileName := NomeArquivo;
53.      ShowDialog := False;
54.    end;
55.    Relatorio.Export(FastPDF);
56.    ShellExecute(0, 'open', PChar(NomeArquivo),nil,nil, SW_SHOWMAXIMIZED);
57.  finally
58.    FreeAndNil(Relatorio);
59.    FreeAndNil(FastPDF);
60.    FreeAndNil(FastDBDataSet);
61.  end;
62. end;
```

IdSSLOpenSSL, deve ser adicionado ao projeto e integrado ao TIdSMTP. Este objeto permite a utilização de SSL pelos componentes Indy. A utilização de SSL é muito indicada, pois, permite que as informações possam ser criptografadas, protegendo desta forma os dados. Outro componente que pode ser utilizado é o TIdMessage da Unit IdMessage. Este componente permite que o conteúdo da mensagem seja elaborado, utilizando diversos recursos como HTML (HyperText Markup Language). Quanto a utilização de anexos, o componente TIdAttachmentFile presente na Unit IdAttachmentFile, pode ser aplicado em conjunto com o componente TIdMessage, permitindo assim o uso deste recurso. Uma vez que os objetos estejam devidamente instanciados e configurados, o envio de e-mail pode ser efetuado. A **Listagem 4** apresenta um exemplo de configuração dos componentes Indy para o envio do documento XML da NF-e por e-mail através do servidor Gmail.

Em um primeiro momento todas as variáveis dos tipos mencionados anteriormente são declaradas entre as linhas 2 e 5. Posteriormente, as classes necessárias são devidamente instanciadas entre as linhas 7 e 9. Algumas propriedades do ClientSmtp são definidas, como é o caso do Host, o endereço SMTP do servidor, bem como o login e a senha do usuário em Username e Password, entre as linhas 10 e 14. Também é definido o número da porta para 587 (linha 15), o número utilizado pelo servidor Gmail, lembrando que a mesma varia de servidor para servidor. Posteriormente, na linha 16, é configurado o manipulador (IOHandler) como SSLHandler para a utilização de TLS (Transport Layer Security), conforme a linha 17. O TLS tem a função de oferecer segurança para o transporte dos dados através de processos criptográficos. Na sequência, entre as linhas 20 e 28, é criada a mensagem (TIdMessage) e definidas as propriedades remetente (From.Address), nome do remetente (From.Name), destinatário (Recipients.EmailAddresses), título (subject) e o conteúdo da mensagem (Body.Add). Logo em seguida é adicionado o documento XML como anexo através do método TIdAttachmentFile.Create, conforme a linha 29. A conexão com o servidor é realizada através do procedimento ClienteSmtp.Connect (linha 32). Para concluir, a mensagem é enviada na linha 37 (ClienteSmtp.Send), o cliente é desconectado do servidor (linha 39) e os objetos são liberados da memória, conforme as linhas 40, 41 e 42.

Listagem 4. Procedimento para o envio do documento XML da NF-e através de E-mail

```
01. procedure TForm.EnviarXMLNFe(ArquivoXML: AnsiString);
02. var
03.   ClienteSmtp: TIdSMTP;
04.   SSLHandler: TIdSSLIOHandlerSocketOpenSSL;
05.   Mensagem: TIdMessage;
06. begin
07.   ClienteSmtp := TIdSMTP.Create(nil);
08.   SSLHandler := TIdSSLIOHandlerSocketOpenSSL.Create(nil);
09.   Mensagem := TIdMessage.Create(nil);
10.  with ClienteSmtp do
11.  begin
12.    Host := 'smtp.gmail.com';
13.    Username := 'login@gmail.com';
14.    Password := 'xxxxxxxx';
15.    Port := 587;
16.    IOHandler := SSLHandler;
17.    UseTLS := utUseRequireTLS;
18.  end;
19.  with Mensagem do
20.  begin
21.    From.Address := 'login@gmail.com';
22.    From.Name := 'FMX NF-e';
23.    Recipients.EMailAddresses := 'destinatario@dominio.com';
24.    Subject := 'Envio de NF-e ' + DateTimeToStr(Now);
25.    Body.Add('Prezado Cliente, '+
26.          'você está recebendo o documento XML da NF-e referente a compra X
             efetuada.'+ 'Obrigado');
27.
28.  end;
29.  TIdAttachmentFile.Create(Mensagem.MessageParts, ArquivoXML);
30.  try
31.    try
32.      ClienteSmtp.Connect;
33.    except
34.      on e:Exception do
35.        ShowMessage('Houve um erro na conexão: '+ e.Message);
36.    end;
37.    ClienteSmtp.Send(Mensagem);
38.  finally
39.    ClienteSmtp.Disconnect;
40.    FreeAndNil(ClienteSmtp);
41.    FreeAndNil(Mensagem);
42.    FreeAndNil(SSLHandler);
43.  end;
44. end;
```

Nota do DevMan

O TCP/IP (Transmission Control Protocol / Internet Protocol) pode ser resumido como um conjunto de diversos protocolos que possibilitam a comunicação entre computadores presentes em uma rede, implementando todos os mecanismos necessários para garantir a integridade das informações transmitidas. Ele também é constituído por diversas camadas que possuem funções específicas.

Conclusão

Diante de vários imprevistos que podem ocorrer durante todo o processo de NF-e, existem outros WebServices que são muito úteis e devem fazer parte da aplicação de nota fiscal eletrônica para torná-la mais completa, como é o caso do processo de emissão em contingência, da inutilização de numeração e do cancelamento da nota.

Como foi possível observar neste artigo, a elaboração do DANFE através do FireMonkey é mais um processo que pode ser realizado sem grandes complicações com a utilização do FastReport. Apesar de não ser destinado a esta plataforma, algumas Units do FastReport podem ser facilmente alteradas para que alguns objetos possam funcionar perfeitamente no FireMonkey, sem a necessidade de qualquer recurso adicional. O envio de e-mails também é outro recurso que pode ser elaborado de maneira simples através do uso da suíte de componentes Indy, possibilitando assim o envio do documento XML da NF-e para o destinatário de uma forma rápida e simples.

De forma geral, foi possível observar que uma aplicação NF-e exige diversos recursos e conhecimentos para ser aplicada, que

vão desde a elaboração das notas fiscais com base nos impostos e processos externos como a obtenção dos certificados, até a utilização de vários recursos do Delphi XE 2 que podem ser aplicados pelo FireMonkey. Vale a pena evidenciar também que este mini-curso apresentou uma simples maneira de criar uma aplicação e, evidentemente, existem outras várias formas que vão de acordo com o desejo ou necessidade de cada desenvolvedor. Foram apresentados ao longo do mini-curso recursos como a manipulação de arquivos XML, XML Data Binding, Certificados, CAPICOM, MSXML, WebServices, FastReport, componentes Indy, dentre outros, demonstrando de certa forma, um pouco de cada um deles.

Giuliano Scombatti Pinto
giuliano@sygnux.com.br
É Analista Desenvolvedor da Sygnux Software (www.sygnux.com.br) localizada em Monte Alto/SP, atuando também com desenvolvimento Web e Android. Formado pela Fatec/TQ, começou a desenvolver com Delphi em 2003. Atualmente trabalha com Delphi, Java, PHP e Flash/Flex. Professor de Informática de Projetos Sociais da Prefeitura Municipal de Monte Alto/SP.

Download do Manual do Contribuinte e dos Schemas
https://www.fazenda.sp.gov.br/nfe/downloads/arquivos_vigentes.asp

Download do CAPICOM
http://www.microsoft.com/downloads/details.aspx?FamilyID=860ee43a-a843-462f-abb5-ff88ea5896f6&displaylang=pt-br

WebDANFE - Impressão de DANFE através da Web
http://www.webdanfe.com.br

Maiores informações sobre o FastReport 4.0
http://www.devmedia.com.br/fastreport-no-delphi-xe2-revista-clubedelphi-137/23300

DevSpace do Autor
http://www.devmedia.com.br/space.asp?id=283274

Dê seu feedback sobre esta edição!

A Clubedelphi tem que ser feita ao seu gosto. Para isso, precisamos saber o que você, leitor, acha da revista!

Dê seu voto sobre este artigo, através do link:

www.devmedia.com.br/clubedelphi/feedback

Conhecimento faz diferença!

Faça já sua assinatura digital ! | www.devmedia.com.br/es

Faça um *upgrade* em sua carreira

Em um mercado cada vez mais focado em qualidade, ter conhecimentos aprofundados sobre requisitos, metodologia, análises, testes, entre ouutros, pode ser a diferença entre conquistar ou não uma boa posição profissional. Sabendo disso a DevMedia lançou uma publicação totalmente especializada em Engenharia de Software. Todos os meses você pode encontrar artigos sobre Metodologias Ágeis; Metodologias tradicionais (document driven); ALM (application lifecycle); SOA (aplicações orientadas a serviçoes); Análise de sistemas; Modelagem; Métricas; Orientação à Objetos; UML; testes e muito mais. **Assine Já!**

DevMedia

Desenvolva para Android
RadPHP e Delphi XE 2 - Parte 1

Giuliano Scombatti Pinto

O atual cenário de desenvolvimento de software tem evoluído muito e está bastante competitivo, uma vez que existem diversos sistemas de alta qualidade direcionados a vários seguimentos do mercado, elaborados por uma gama de Software Houses competentes. Diante deste fator, os desenvolvedores precisam oferecer a seus clientes aplicações cada vez mais ricas e inovadoras, que forneçam um diferencial com relação a seus competidores e os ambientes que apresentam grande potencial são as áreas Web e móvel. Há muito tempo a plataforma Web é um ambiente extremamente importante no dia-a-dia de diversos softwares e consequentemente seus usuários, uma vez que permite a implementação de diversos recursos como WebServices, as aplicações destinadas a receber, processar e responder requisições de forma automatizada através de uma rede como a Internet. Quanto ao ambiente móvel, é possível notar a grande força do mesmo observando suas diversas tecnologias suportadas, evidenciando também a constante evolução e tendência de mercado. O recurso de touch-screen presente nos Smartphones, Tablets, etc, se consolidou e se popularizou entre os usuários, tornando-se uma importante peça no desenvolvimento de aplicações cada vez mais interativas. As pessoas podem utilizar seus dispositivos móveis para navegar na Internet, ouvir músicas, trocar informações, jogar, efetuar ligações, etc, o que os tornam mini-computadores prontos para efetuar uma ação a qualquer momento e de forma remota, diversos pontos que viabilizam o investimento neste ambiente. Diante deste contexto, houve grande popularização dos sistemas operacionais móveis Android, iOS, Windows Mobile, dentre outros. O Android é uma importante plataforma da atualidade que merece destaque por ser aberta, flexível e conter uma diversidade de aplicações, evidenciando que a mesma está presente em milhões de dispositivos ao redor do mundo. Sendo assim, nos dias de hoje existem diversas ferramentas e frameworks que permitem que os desenvolvedores possam elaborar aplicações nativas para o mesmo.

Além do Delphi, a suíte de ferramentas RAD Studio XE 2 também é composta por outras aplicações importantes que abrem novos horizontes para o desenvolvimento de sistemas mais complexos e uma que merece destaque é o RadPHP. Baseado na linguagem PHP, muito robusta e orientada a objetos, esta ferramenta permite que os desenvolvedores já acostumados com o ambiente e os diversos recursos de alta produtividade oferecidos pelo Delphi, tal como o "arrastar e soltar", utilizem ela para desenvolver aplicações completas para a plataforma Web ou móvel de forma fácil, organizada e rápida.

Uma das principais vantagens ao utilizar o RadPHP é a possibilidade de exportar a aplicação de diretamente para diversas plataformas móveis presentes no mercado, utilizando o mesmo código fonte. Em um primeiro momento, as aplicações podem ser testadas diretamente em um navegador de Internet como o Chrome, IE, Firefox, etc, o que aumenta a agilidade na fase de testes,

Resumo DevMan

De que trata o artigo:

De maneira geral este mini-curso abordará a integração entre uma aplicação DataSnap (servidor) desenvolvida no Dephi XE 2 com uma aplicação cliente desenvolvida para a plataforma Android através do RadPHP, uma ferramenta do RAD Studio XE 2 para o desenvolvimento de aplicações através da linguagem PHP. Esta primeira parte traz uma introdução a esta poderosa ferramenta de desenvolvimento, seus principais recursos e suas grandes possibilidades ao que concerne a elaboração de aplicações que podem ser exportadas para diversas plataformas, bem como apresenta um simples exemplo prático para o RadPHP.

Em que situação o tema é útil:

Possibilitar a integração entre uma aplicação desenvolvida para a plataforma móvel com um servidor DataSnap Delphi para o compartilhamento de informações, bem como a familiarização com a ferramenta RadPHP.

Desenvolvendo para Android:

O sistema operacional Android, assim como a plataforma móvel em geral, vem se tornando mais popular e presente na vida das pessoas, oferecendo diversos recursos como multimídia, aplicações, jogos, etc. Diante destes fatores, este sistema é um ambiente potencial para o desenvolvimento e integração de softwares comerciais. Sendo assim, uma ótima ferramenta para ser utilizada no projeto de aplicações para esta plataforma é o RadPHP XE 2. Através dele, é possível gerar e integrar aplicações para diversas plataformas móveis com base no PHP, uma linguagem de programação orientada a objetos. Este artigo tem o intuito de apresentar a ferramenta RadPHP, suas características e principais recursos, demonstrando a facilidade e a agilidade que é obtida no desenvolvimento de projetos para a plataforma móvel, bem como um simples exemplo de como utilizar esta ferramenta.

não necessitando diretamente de um emulador para a plataforma alvo, uma vez que o mesmo tende a ser mais lento.

Uma introdução à linguagem PHP e suas características

A linguagem de programação server-side PHP (PHP: Hypertext Preprocessor) teve sua concepção na década de 90, sendo uma plataforma amplamente utilizada no ambiente de desenvolvimento Web em conjunto com a linguagem HTML (HyperText Markup Language) por ser rápida, multi-plataforma, robusta e open source. Também recebe a denominação de linguagem de script e sua sintaxe é muito semelhante a das linguagens Java e C, sendo orientada a objetos. O PHP tem por meta possibilitar a geração de páginas HTML de uma forma mais rápida e inteligente, podendo utilizar uma variedade de bancos de dados e outros recursos, mas com o passar do tempo se tornou uma linguagem completa que pode ser aplicada para diversos fins, inclusive o desenvolvimento de aplicações desktop. Por ser executada no servidor (server-side), o usuário da página não tem acesso ao código fonte do mesmo, mas somente ao script HTML que foi gerado por ele. Uma característica importante desta linguagem é que suas variáveis recebem o prefixo $, levando em consideração que são case-sensitive (distinção entre maiúsculas e minúsculas). Apesar de ser uma boa prática, a inicialização de variáveis não é uma obrigatoriedade nesta linguagem. Outro ponto é com relação às variáveis pré-definidas, onde muitas são superglobais (elementos disponíveis para todos os escopos) e permitem a obtenção e manipulação de diversos dados. Quanto aos tipos, o PHP suporta boolean, integer, float, string, array, object, resource e NULL. Com relação a palavras reservadas, o PHP possui algumas características particulares. A **Tabela 1** demonstra algumas das principais palavras reservadas e variáveis pré-definidas do PHP.

Outro fator de destaque que também deve ser considerado é com relação às instruções e estruturas de controle. O PHP possui diversas estruturas condicionais, de laço, entre outras, tais como if, else, elseif, while, for, foreach, switch, goto, etc, que fornecem uma base para a elaboração dos blocos de códigos. Evidentemente, diversos operadores podem ser aplicados as estruturas de controle. A **Tabela 2** apresenta os principais operadores utilizados na linguagem PHP.

Palavras e variáveis pré-definidas	Descrição
<?php	Utilizado para iniciar o desenvolvimento dos códigos em PHP.
?>	Utilizado para indicar o final do código PHP.
$this	Utilizado para se referir ao objeto atual.
$GLOBALS	Variáveis presentes no escopo global.
$_SERVER	Utilizada para a obtenção de informações do servidor, uma vez que esta linguagem é server-side.
$_GET	Utilizado para a manipulação de valores presentes nas variáveis do método HTTP GET, muito utilizado para a transferência de informações entre páginas.
$_POST	Utilizado para a manipulação das variáveis submetidas através do método HTTP POST, também utilizado para a transferência de dados entre uma página e outra.
$_FILES	Utilizada para a manipulação de arquivos enviados.
$_SESSION	Utilizada para a manipulação de variáveis de seção. Tais variáveis também permitem a troca de informações entre as páginas da aplicação.
$_COOKIES	Utilizada para a manipulação dos Cookies, as informações que ficam gravadas em um dispositivo de armazenamento.

Tabela 1. Principais palavras reservadas e variáveis pré-definidas do PHP

Operadores	Descrição
and, &&	Operador lógico “E”, verdadeiro se ambas as expressões são verdadeiras
or, ||	Operador lógico “Ou”, verdadeiro se uma das expressões é verdadeira.
xor	Operador lógico “Ou exclusivo”, verdadeiro se uma das expressões é verdadeira, mas não ambas.
!	Operador lógico de negação (not), verdadeiro se a expressão não é verdadeira.
+, -, *, /, %	Aritméticos, adição, subtração, multiplicação, divisão e módulo.
++, --	Operadores para incremento e decremento de valores.
=, +=, -=	Atribuição
==	Comparação, igualdade.
!=, <>	Comparação, diferente.
<, <=	Comparação, menor que e menor ou igual.
>, >=	Comparação, maior que e maior ou igual.
.	Concatenação.

Tabela 2. Principais operadores da linguagem PHP

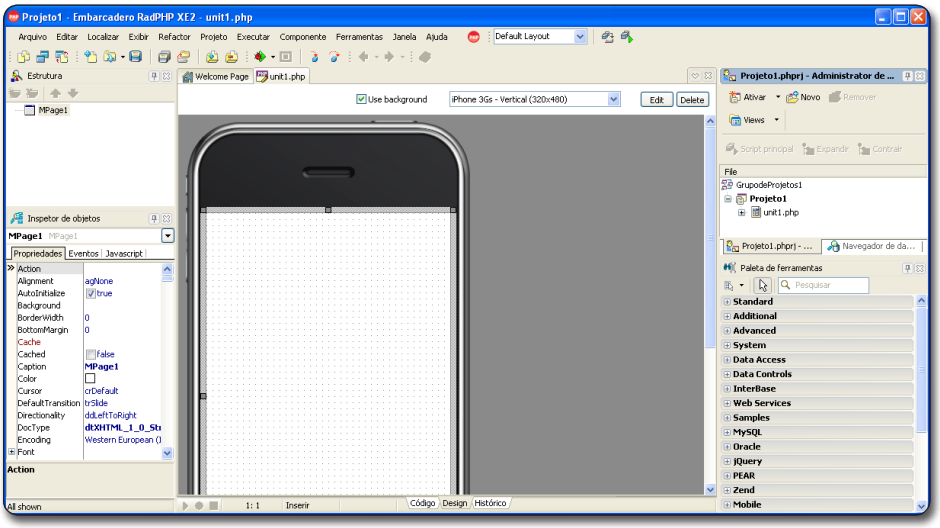


Figura 1. Ambiente de desenvolvimento do RadPHP XE 2

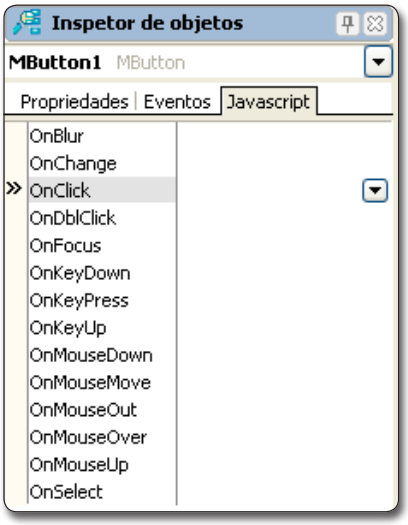


Figura 2. Inspetor de Objetos com os eventos da aba JavaScript

A ferramenta RadPHP

A ferramenta RadPHP possibilita unir todos os recursos e vantagens da linguagem PHP a um editor visual de alta produtividade, provendo uma sincronia entre o design gráfico da aplicação ou página e o código fonte, possuindo também uma versão em Português, sem contar que permite uma integração facilitada com algumas aplicações desenvolvidas no ambiente Delphi. Esta ferramenta também engloba as tecnologias HTML5, CSS3, JavaScript, jQuery, entre outros. O HTML5 é a quinta versão da linguagem HTML, a linguagem padrão para a definição e exibição das páginas de Internet, na qual recebeu diversos recursos como a adição de novas Tags nesta versão. O CSS3 (Cascading Style Sheets) também é uma tecnologia importante da Web que evolui constantemente e tem por objetivo principal a definição de padrões para diversos estilos e efeitos para as páginas HTML e seus elementos. O JavaScript é outra tecnologia importante para o desenvolvimento de páginas HTML, sendo uma linguagem baseada em scripts e muito semelhante ao Java e o C++. Ao contrário do PHP, o JavaScript é client-side (executado na aplicação cliente), o que implica que seu código fonte pode ser visualizado pelo o usuário. Através dele, é possível adicionar diversos recursos como pré-validações de campos, animações, efeitos, etc. Já a jQuery é um poderoso framework baseado na linguagem JavaScript que provê diversas funcionalidades para as páginas HTML, como animações, transições e manipulação de eventos, entre outros, de uma forma rápida e simplificada, uma vez que possui diversos métodos prontos.

O ambiente do RadPHP se assemelha muito ao Delphi, contendo a visualização da Estrutura, o Inspetor de Objetos, a área de desenvolvimento, o Administrador de Projetos e a Paleta de Ferramentas. A **Figura 1** apresenta o ambiente visual de desenvolvimento do RadPHP XE 2.

A janela do Inspetor de Objetos, além de trazer as abas Propriedades e Eventos que pertencem aos componentes, também presentes no Delphi XE 2, conta ainda com uma aba denominada JavaScript. Através desta aba o desenvolvedor pode inserir códigos em JavaScript para que sejam executados de acordo com o evento em questão. Em casos onde um mesmo evento é preenchido para ambas as abas (Eventos e JavaScript), como o OnClick por exemplo, são executados os códigos de cada evento, porém vale ressaltar que o evento OnClick do JavaScript é ideal para efetuar validações. A **Figura 2** apresenta o Inspetor de Objetos com os eventos da aba JavaScript.

Outro ponto do ambiente de desenvolvimento da ferramenta que merece destaque é a janela Paleta de ferramentas. Através dela, é possível inserir diversos componentes visuais e não visuais para o projeto PHP, e duas paletas presentes nesta janela merecem um maior destaque, sendo elas a Mobile e Mobile Hardware. Na paleta Mobile se encontram os componentes visuais destinados à plataforma móvel, que por sua vez apresentam um visual mais refinado e familiarizado para o ambiente. Já a paleta Mobile Hardware traz consigo diversos componentes voltados para a manipulação dos vários recursos da plataforma móvel, tais como a câmera, o acelerômetro, os arquivos, entre outros. A **Figura 3** apresenta a janela "Paleta de ferramentas" padrão do RadPHP, que contém diversos componentes, destacando a paleta Mobile Hardware.

Diante de tantos recursos, a ferramenta RadPHP permite a elaboração de diversos projetos e arquivos. A **Figura 4** demonstra a janela "Novos Itens", utilizada para a criação de um novo projeto ou adição de itens ao mesmo.

A opção "Aplicação Facebook" é um novo recurso que permite o desenvolvimento de uma aplicação voltada para o site de relacionamentos Facebook. Este tipo de projeto é baseado em uma aplicação Web comum, porém incorpora recursos para a comunicação com tal site, permitindo assim a obtenção de dados como conteúdo e informações dos usuários, devendo tal aplicação ser futuramente hospedada em um servidor.

Ao utilizar a opção "Aplicação RPCL", o desenvolvedor tem seu foco na elaboração de aplicações baseadas na arquitetura

RPCL (RadPHP Component Library - Biblioteca de Componentes do RadPHP). Ela provê um grupo de componentes visuais e não visuais com seus devidos eventos que podem ser utilizados para a rápida elaboração de aplicações baseadas em PHP. Através deste recurso é possível criar aplicações com uma interface gráfica contendo conexão com bancos de dados, aplicações Web, aplicativos de console, etc.

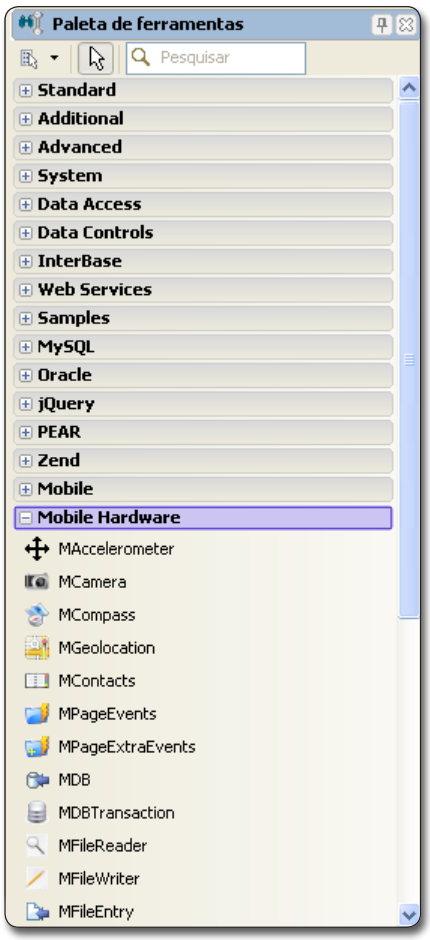


Figura 3. Janela Paleta de ferramentas evidenciando a paleta Mobile Hardware

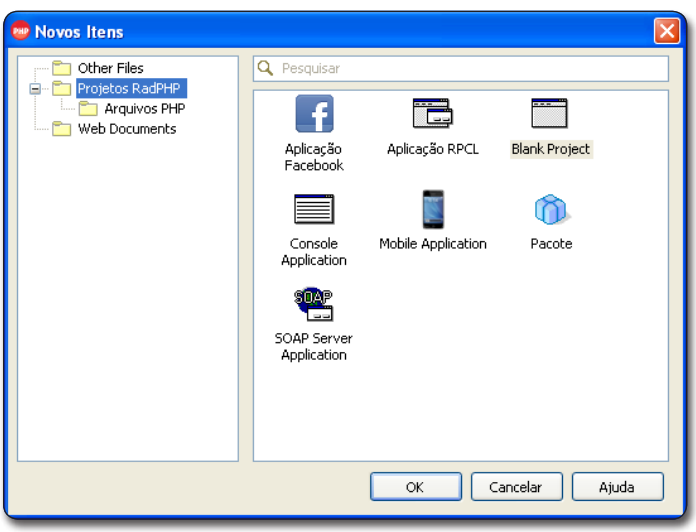


Figura 4. Janela "Novos Itens" para a criação de um projeto no RadPHP

Como seu próprio nome sugere, a opção "SOAP Server Application" permite a elaboração de uma aplicação Web baseada no protocolo SOAP (Simple Object Access Protocol), que por sua vez é um padrão adotado para a troca de informações entre o WebService e a aplicação cliente.

De fato, uma das opções que mais chama a atenção no RadPHP é a "Mobile Application", uma vez que esta é direcionada a elaboração de aplicações para a plataforma móvel. Com um mesmo projeto, por exemplo, é possível exportar a aplicação para o sistema Android, presente em diversos Smartphones e Tablets, ou iOS, presente nos iPhones, iPads e iPods, utilizando os recursos da plataforma PhoneGap. Esta plataforma por sua vez permite a criação de aplicações diretamente para diversos sistemas operacionais móveis através das tecnologias utilizadas na Web, tais como HTML5, CSS3 e JavaScript, tendo também a vantagem de ser um projeto de código fonte livre em constante evolução.

Diante destes diversos recursos, uma vez que a aplicação esteja concluída e testada, basta utilizar o assistente de exportação para a plataforma móvel, presente no menu Ferramentas>Wizard for PhoneGap, para efetuar o processo de geração da aplicação nativa para o sistema operacional alvo. A **Figura 5** apresenta a janela do assistente de exportação, "Wizard for PhoneGap".

Nota do DevMan

O PhoneGap é um framework free que permite o desenvolvimento nativo e multiplataforma para dispositivos móveis utilizando como linguagem de programação o HTML5 e javascript. Ele disponibiliza através do javascript acesso nativo aos recursos de hardware dos dispositivos, sendo compatível com vários modelos como Windows Phone, Android, IOS (IPhone, IPad), Blackbery, Symbian e outros mais. Basicamente seu funcionamento se dá na seguinte forma: você constrói seu aplicativo, sendo a interface em HTML5 e a lógica toda em javascript. Como isso tudo é web, você pode testar em um browser qualquer. Estando tudo certo, você realiza o download do PhoneGap para a plataforma que você precisa. Por exemplo Android. Então esse framework é instalado dentro do IDE de desenvolvimento para Android (Eclipse) e nele você cria um novo projeto PhoneGap. Nesse novo projeto você inclui suas páginas HTML5, adiciona seus scripts e compila o aplicativo que já estará pronto para ser testado no emulador e posteriormente ser distribuído. O segredo todo está no fato de que o PhoneGap encapsula o navegador web de cada plataforma, de tal forma que o usuário final nem percebe que na verdade está executando uma página local.

De fato, como pôde ser observado, o RadPHP é uma ferramenta de alta produtividade e flexível, que automatiza e facilita o trabalho do desenvolvedor através de seus assistentes de importação e exportação, permitindo a elaboração de aplicações robustas para diversas plataformas de uma maneira fácil.

O sistema Android e o RadPHP

O Android é um sistema operacional muito popular da atualidade, englobando diversas tecnologias e recursos que podem ser utilizados pelos desenvolvedores para elaborar desde simples aplicações à complexos jogos 3D. Ao obter a instalação do RadPHP, o desenvolvedor também conta com a instalação de

todos os softwares necessários para o desenvolvimento destinado a esta plataforma, através do executável "android-setup.exe". Com ele é possível instalar o Java JDK (Java SE Development Kit), o Android SDK (Software Development Kit) e o Apache-Ant. O Java, além de ser uma linguagem de programação, oferece a plataforma base para o funcionamento de diversos recursos e programas, o Android SDK traz consigo diversas ferramentas para o desenvolvimento de aplicações para esta plataforma móvel e o Apache-Ant é uma biblioteca baseada em Java e linha de comando que possui vários recursos para auxiliar no desenvolvimento de softwares. Após a instalação, o desenvolvedor pode personalizar ou utilizar as ferramentas do Android através das aplicações presentes nos diretórios tools e platform-tools, ambos presentes no diretório "C:\Arquivos de programas\Android\android-sdk\", sendo assim, para personalizar o emulador ou criar outros, basta acessar o arquivo de lote android.bat. O emulador é um recurso importante presente no conjunto de ferramentas do Android pois permite que a aplicação seja testada para esta plataforma sem a necessidade de um dispositivo físico, possibilitando que o desenvolvedor verifique o comportamento que o software terá neste ambiente.

Nota do DevMan

O arquivo de Lote (.bat) é constituído por uma sequência de um ou mais comandos e seus atributos com o intuito de automatizar alguma tarefa, como o carregamento de um programa, uma rotina de Backup (cópia de segurança), etc. Foi muito utilizado no antigo sistema operacional MS-DOS e continua sendo até os dias de hoje, podendo ser elaborado através de um simples editor de textos ou através do comando copy con no prompt de comandos.

RadPHP na prática

Para dar início a uma maior familiarização com o RadPHP será criado um simples exemplo de aplicação para consumir um WebService de clima. Para tanto, o WebService que será utilizado para este exemplo é um serviço gratuito para a consulta do clima de diversas cidades do mundo, presente no site http://www.webservicex.net. Em um primeiro momento, o WebService é acessado para obter as cidades disponíveis para a consulta tendo seu conteúdo atribuído a um ComboBox e, em uma segunda etapa, ele é acessado novamente para efetuar a consulta do clima de acordo com a cidade selecionada pelo usuário. Sendo assim, deve-se criar um novo projeto de aplicação RPCL através do menu Arquivo>Novo>Aplicação RPCL - PHP, sendo a unit gravada com o nome de unitPrincipal.php e o formulário recebeu a denominação de "Pagina". Para este exemplo, foram inseridos três Labels, um ComboBox, um Edit e um Button. Na sequência, foi criado o código para o evento OnCreate da Página, executado no momento da criação da mesma, com o intuito de consumir o WebService obtendo as cidades disponíveis para a consulta do clima no Brasil. A **Listagem 1** apresenta o código do evento OnCreate para o primeiro consumo do WebService.

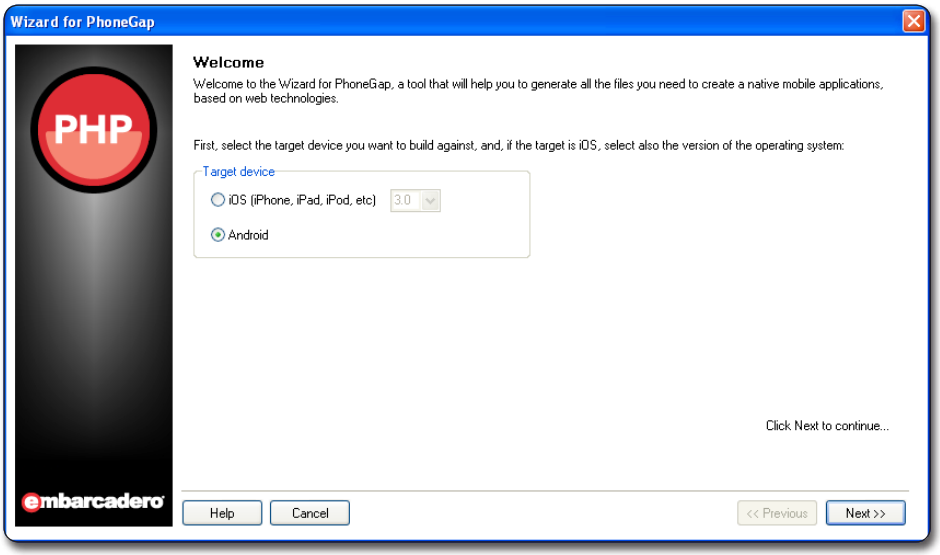


Figura 5. Janela do assistente de exportação "Wizard for PhoneGap"

Listagem 1. Código do evento OnCreate

```
function PaginaCreate($sender, $params)
{
    $url = 'http://www.webservicex.net/globalweather.asmx?wsdl';
    $cliente = new SoapClient($url, array('encoding'=>'UTF-8'));
    $resultado = $cliente->GetCitiesByCountry(array('CountryName'=>'Brazil'));
    $xml = simplexml_load_string($resultado->GetCitiesByCountryResult);
    foreach ($xml->Children() as $no)
      $this->cbCidade->AddItem((string)$no->City);
}
```

Neste trecho de código é possível notar que é atribuído o endereço do WebService para a variável $url. Posteriormente é criado um objeto do tipo SoapClient com os parâmetros endereço ($url) e codificação (UTF-8), tendo seu valor armazenado na variável $cliente. A classe SoapClient possui diversos recursos que permitem o retorno dos métodos presentes em um WebService. A variável $resultado recebe a resposta do consumo do WebService através do método GetCitiesByCountry(), cujo objetivo é obter todas as cidades disponibilizadas para a consulta do clima. Neste método é possível observar que é passado o parâmetro "CountryName" como "Brazil" em forma de array para o funcionamento do método no trecho "array('CountryName'=>'Brazil'). Para auxiliar na obtenção dos métodos caso o desenvolvedor não conheça o WebService, ele pode utilizar em tempo de desenvolvimento a função var_dump($cliente->__getFunctions()). A função var_dump() é extremamente útil

pois permite o retorno do valor presente em uma variável. Na sequência a variável $xml recebe o resultado da função simplexml_load_string() que tem como parâmetro o resultado retornado pelo WebService (GetCitiesByCountryResult). Esta função permite o carregamento de uma string XML em um objeto. Para concluir, é utilizado um foreach (para cada) com o intuito de acessar cada nó filho do XML ($xml->Children()), recebendo a denominação de $no e o ComboBox (cbCidade) adiciona em sua lista de item (AddItem) o conteúdo em string do nó City ((string) $no->City), efetuando o cast (conversão) do mesmo.

O próximo passo no projeto é a elaboração do código que efetuará a consulta quando o usuário selecionar a cidade e clicar no botão, apresentando o resultado da mesma. Sendo assim, foi adicionado um novo formulário ao projeto (menu Arquivo>Novo>Formulário PHP) que foi denominado unitExibirResultado.php. Neste formulário foram adicionados apenas três Labels para a exibição (lbCidade, lbInfo e lbTemperatura). Retornando ao formulário principal, foi adicionado o código necessário para o botão que foi denominado "Consultar". A **Listagem 2** demonstra o código que foi adicionado ao mesmo com o intuito de consumir o WebService e apresentar a resposta em outro formulário.

Neste trecho de código, a variável $url recebe o endereço do WebService. Também foi criada uma nova variável ($selecionado) para o armazenamento da opção selecionada pelo usuário (ItemIndex) no ComboBox (cbCidade). Posteriormente, a variável $cidade recebe a cidade selecionada na lista do ComboBox (Items[$selecionado]). Uma nova variável $pais também é utilizada para o armazenamento do país. Sendo assim, a obtenção dos métodos é efetuada através do novo objeto SoapClient que teve seu conteúdo armazenado na variável $cliente. Na sequência, a variável $resultado é utilizada para armazenar a resposta obtida com a execução do método GetWeather(). Tal método tem seus parâmetros (array) definidos de acordo com as variáveis $cidade e $pais. Posteriormente a variável $xml recebe o carregamento em XML do retorno do WebService presente na variável $resultado. Neste momento também é utilizada a função preg_replace(), cujo objetivo é substituir determinado padrão por um outro valor, onde neste caso o padrão "/utf-16/", presente na mensagem de retorno do WebService (GetWeatherResult), é substituída por "utf-8" para a correta criação do XML. Para concluir, o outro formulário que foi denominado "unitExibeResultado.php" é chamado através de redirect(), cuja função é efetuar o redirecionamento. Também é possível notar que este formulário ou página é chamado contendo algumas variáveis sendo passadas através do método GET, uma vez que este recurso é uma das formas de transferir dados entre páginas. Para efetuar a transferência de valores com o uso de GET, basta adicionar ao final do endereço da página o caractere "?" e informar o nome da variável, como neste exemplo, "cid", adicionando também seu conteúdo com o uso do sinal "=". Vale à pena ressaltar que a cada nova variável que for necessária, basta inserir o caractere "&" e assim preencher conforme desejado.

Neste momento os códigos do formulário principal são encerrados, restando apenas inserir o código para o formulário unitExibeResultado.php, para que o mesmo possa exibir o retorno do WebService através dos três Labels adicionados anteriormente. A **Listagem 3** apresenta o código utilizado no evento OnCreate do outro formulário para o carregamento das informações através de GET.

Neste simples trecho de código, cada Label é preenchido (Caption) de acordo com as variáveis que são obtidas com o uso da variável PHP pré-definida $_GET, cuja função é possibilitar o carregamento de variáveis transmitidas através do método GET.

Como pôde ser observado, a linguagem PHP com o uso do Rad PHP permite que tarefas como o consumo de um WebService possam ser realizadas de uma maneira direta, rápida e fácil.

Listagem 2. Código do evento OnClick do Botão Consultar

```
function ConsultarClick($sender, $params)
{
    $url = 'http://www.webservicex.net/globalweather.asmx?wsdl';
    $selecionado = $this->cbCidade->ItemIndex;
    $cidade = $this->cbCidade->Items[$selecionado];
    $pais = $this->edPais->Text;
    $cliente = new SoapClient($url, array('encoding'=>'UTF-8'));
    //var_dump($cliente->__getFunctions());
    $resultado = $cliente->GetWeather(array('CityName'=>$cidade,'CountryName'=>$pais));
    $xml = simplexml_load_string(preg_replace('/utf-16/', 'utf-8', $resultado->GetWeatherResult));
    redirect('unitExibeResultado.php?cid='. $xml->Location .
    '&info='. $xml->Time . '&temp='. $xml->Temperature);
}
```

Listagem 3. Código do evento OnCreate da unitExibeResultado.php

```
function ExibeResultadoShow($sender, $params)
{
    $this->lbCidade->Caption = 'Cidade = '. $_GET['cid'];
    $this->lbInfo->Caption = 'Hora = '. $_GET['info'];
    $this->lbTemperatura->Caption = 'Temperatura = '. $_GET['temp'];
}
```

Uma breve introdução ao DataSnap

Para criar uma aplicação integrada entre ambas as plataformas, Windows e Android, o desenvolvedor pode utilizar os recursos de um servidor DataSnap. Sendo assim, este é o recurso que será utilizado nos próximos artigos deste mini-curso. Ela é uma tecnologia presente no Delphi que permite o desenvolvimento de softwares multicamadas muito utilizada em integrações com banco de dados. Sendo assim, ele fornece a base para a criação de aplicações cliente-servidor que podem ser disponibilizadas através da Internet ou uma rede qualquer. Dentre as características desta tecnologia, podem-se citar a capacidade da invocação de métodos remotos por uma aplicação cliente, a possibilidade da comunicação entre duas aplicações desenvolvidas na plataforma Delphi, C++, RadPHP, entre outros.

Dentro deste cenário, existem alguns componentes que merecem destaque, sendo eles o TDSServer e o TDSServerClass. O TDSServer é o componente principal

Nota do DevMan

O método GET, presente no HTTP (HyperText Transfer Protocol), é uma das bases para a transferência de dados entre páginas da Internet, sendo também uma das formas mais práticas. Ele pode ser utilizado através de um formulário (<form>) para o envio das informações através do atributo method=GET. Quando aplicado, o fluxo que pertence aos dados é separado do endereço através do caractere "?", sendo assim, o desenvolvedor também pode inserir as informações manualmente desta maneira.

para a aplicação servidor DataSnap, evidenciando também que tal aplicação necessita apenas de um objeto deste tipo. Através dele é efetuado o gerenciamento do servidor, sendo possível a realização de funções como inicialização e encerramento. Quanto ao TDSServerClass, é o componente utilizado para que seja especificada uma classe de servidor contendo os métodos que podem ser invocados remotamente por uma aplicação cliente.

Outro fator que também é importante dentro do desenvolvimento de um servidor DataSnap é a tecnologia REST (Representational State Transfer – Transferência de estado representacional), uma recente arquitetura para a elaboração de WebServices que permite o compartilhamento e mesclagem de diversas informações, permitindo que aplicações fracamente acopladas sejam desenvolvidas, que também esta presente no RadPHP.

Conclusão

O RadPHP é sem dúvida uma importante e potencial ferramenta para o desenvolvimento de aplicações Web ou destinadas à diversas plataformas móveis de uma maneira rápida e simplificada através de suas ferramentas e componentes.

Como foi possível observar neste artigo, o PHP é uma linguagem de programação robusta, flexível e aberta, sendo fácil de ser entendida por um desenvolvedor já habituado com a linguagem Java e C, e ao ser aliada a ferramenta RadPHP, permite o desenvolvimento de aplicações completas para diversos ambientes.

Nesta primeira parte deste mini-curso o objetivo foi apresentar a ferramenta e suas facilidades, bem como um simples exemplo para iniciar uma familiarização com a mesma, oferecendo assim uma base para que nos próximos artigos seja iniciado o projeto de uma aplicação DataSnap desenvolvida no Delphi XE 2 com acesso a banco de dados para prover a integração com uma aplicação cliente no sistema Android.

Giuliano Scombatti Pinto
giuliano@sygnux.com.br
É Analista Desenvolvedor da Sygnux Software (www.sygnux.com.br) localizada em Monte Alto/SP, atuando também com desenvolvimento Web e Android. Formado pela Fatec/TQ, começou a desenvolver com Delphi em 2003. Atualmente trabalha com Delphi, Java, PHP e Flash/Flex. Professor de Informática de Projetos Sociais da Prefeitura Municipal de Monte Alto/SP.

Site oficial e manual do PHP
http://www.php.net/manual/pt_BR/

Maiores informações sobre PHP
http://www.w3schools.com/php/

DevSpace do Autor
http://www.devmedia.com.br/space.asp?id=283274

Dê seu feedback sobre esta edição!

A Clubedelphi tem que ser feita ao seu gosto. Para isso, precisamos saber o que você, leitor, acha da revista!

Dê seu voto sobre este artigo, através do link:

www.devmedia.com.br/clubedelphi/feedback

Herança Aplicada

Utilizando herança nas aplicaçõe

O conceito de Orientação a Objetos (OO) está presente em grande parte das linguagens de programação da atualidade e possibilita o projeto e desenvolvimento de softwares de uma maneira organizada, sendo baseado em coleções de objetos que possuem seus próprios comportamentos e estruturas. Teve seus primeiros conceitos oficializados na década de 60 pelo Simula, uma linguagem que incorporou diversas facilidades para a definição de classes, subclasses e objetos. No cenário da OO, o planejamento das aplicações tem seu foco voltado para os objetos do mundo real, sua implementação computacional, suas informações e suas funções.

Existem diversas vantagens ao aplicar este conceito, como a reutilização ou compartilhamento de códigos, redução do custo, aumento da produtividade, qualidade, maior confiabilidade, facilidade de manutenção, entre outros.

Na OO são implementadas as classes para a definição dos objetos que compõem a aplicação, onde cada classe determina os comportamentos e estados dos objetos, bem como seus relacionamentos. Um importante recurso presente neste conceito é a herança. Ela é extremamente útil no desenvolvimento de aplicações e um dos pilares da Orientação a Objetos. O modelo convencional deste recurso prevê que as classes podem herdar métodos e atributos de outras, denominadas superclasses ou classes "pai", ou de uma hierarquia, onde as classes que herdam as informações de outras são conhecidas como derivadas, descendentes, classes filhas ou subclasses. Estas últimas por sua vez, podem implementar características únicas e particulares além das propriedades herdadas da classe pai, frisando que tais propriedades não precisam ser repetidas em cada subclasse. O conceito de hierarquia prevê que uma classe mais genérica, ocupando o topo, vai sendo estendida e as classes que se encontram na base são mais especificas. Este conceito é muito importante para a organização e ordenação de classes em projetos composto por um extenso conjunto de objetos.

Outro ponto importante com relação aos atributos e métodos de uma classe é o conceito de controle de acesso. Ele prevê que tais elementos são declarados dentro de uma seção, também conhecida como especificador de acesso. O especificador pode ser privado, público ou protegido. No especificador privado (private), a visibilidade dos elementos como atributos e métodos é limitada, implicando que os mesmos são exclusivamente acessados pelas operações internas da própria classe. No especificador protegido (protected) os membros da classe base tornam-se elementos protegidos da classe derivada. Já no especificador público (public) os membros são visíveis por outras classes, permitindo assim que sejam efetuados acessos externos a tais recursos sem restrições.

Existem outros conceitos presentes na OO que também merecem destaque, entre eles a agregação e a composição.

Resumo DevMan

De que trata o artigo:

Este artigo apresenta uma simples introdução a alguns conceitos importantes da Orientação a Objetos que serão aplicados ou comentados, tendo seu foco voltado para a utilização do recurso de herança, que em suma provê a capacidade de uma classe herdar outra, demonstrando formas de aplicar a mesma no Delphi com formulários e outros componentes.

Em que situação o tema é útil:

Aumentar a produtividade no desenvolvimento e manutenção de qualquer software personalizado aplicando alguns conceitos de OO como herança, bem como demonstrar uma forma de sobrescrever, estender ou adicionar métodos a classes existentes, possibilitando assim a alteração do comportamento das mesmas, além de proporcionar uma melhor organização dos códigos do projeto.

Herança Aplicada:

A Orientação a Objetos em suma, tem por base a utilização de grupos de objetos contendo atributos e métodos que podem interagir entre si. Este conceito é amplamente utilizado pelas ferramentas de desenvolvimento de aplicações da atualidade, incluindo o Delphi XE 2 e outras versões. Com recursos como a herança, o desenvolvimento de um software se torna uma tarefa mais simples e rápida, uma vez que os elementos e códigos das classes podem ser reaproveitados de uma forma mais eficiente, possibilitando assim que métodos sejam alterados, sobrescritos, sobrecarregados, estendidos, etc. Além disso, também é possível economizar muito tempo nas manutenções, uma vez que em vários casos as informações encontram-se centralizadas em uma classe pai, que uma vez alterada, faz com que todas as subclasses descendentes da mesma recebam a atualização. Neste artigo é apresentado como utilizar a herança na prática nas aplicações Delphi para aumentar a produtividade e economizar muito tempo no desenvolvimento de qualquer aplicação que utilize elementos e métodos em comum.

s Delphi

Giuliano Scombatti Pinto

O termo agregação define que existe uma fraca dependência entre um objeto com outros objetos (partes), onde a remoção de um objeto não implica que o outro também seja removido. Já no caso da composição um objeto é composto por outros, o que implica que se o mesmo for removido, todos os outros também serão eliminados, definindo assim uma dependência forte. A **Tabela 1** apresenta outros termos importantes muito utilizados na Orientação a Objetos que serão comentados neste artigo ou que são de muita importância.

Conceitos de herança no Delphi

Assim como outros ambientes de desenvolvimento, o Delphi também possui diversas formas de se aplicar o conceito de herança (Inheritance) desde versões anteriores a XE 2. O mesmo tem seu funcionamento baseado em classes e objetos que podem ser facilmente herdados e estendidos de uma maneira simples para serem adaptados às necessidades do projeto atual ou de acordo com o desejo do desenvolvedor. O Delphi conta com um comando denominado inherited, onde sua utilização sem qualquer informação adicional permite que seja efetuada uma chamada ao método que possui o mesmo nome que o atual na classe pai. A **Tabela 2** apresenta alguns tipos e métodos presentes no Delphi que possuem uma relação e são amplamente utilizados com o recurso de herança.

Existem alguns comandos mencionados que merecem um maior destaque quando a herança será aplicada no projeto, além do Inherited já comentado, sendo eles o virtual, overload e override, onde os mesmos são aplicados em conjunto com a declaração de uma função ou procedimento. O virtual, por sua vez, deve ser aplicado na superclasse com o objetivo de permitir que o método em questão seja sobrescrito ou estendido pelas classes descendentes. Sendo assim, através do uso do comando override na subclasse, tal método é facilmente alterado. Com relação ao overload, ele permite a utilização do polimorfismo em Delphi, implementando o conhecido recurso de sobrecarga de métodos da OO onde as funções e procedimentos de mesmo nome recebem parâmetros diferentes e podem possuir comportamentos distintos.

Os métodos construtores e destrutores também são muito importantes quando o assunto é estender as funcionalidades e propriedades. O método construtor de uma classe sempre é

Conceitos	Descrição
Classe	Uma classe pode ser definida como a descrição de um conjunto de objetos com suas características, comportamentos e relacionamentos semelhantes. Ela pode ter diversas informações abstraídas, apresentando somente as que são relevantes.
Objeto	Um objeto pode ser definido como uma entidade que pode possuir diversas propriedades, comportamentos e uma identidade. Tem o propósito de possibilitar o entendimento do mundo real e oferecer o suporte necessário para a implementação computacional. Também pode ser definido como uma instância de uma classe.
Atributo	Os atributos são utilizados para armazenar as informações de um Objeto, em outras palavras, possibilitam a representação da sua estrutura. Eles podem receber valores distintos e possuem suas definições efetuadas na classe.
Métodos	O método pode ser definido como um comportamento realizado por um objeto, como determinada função e também deve ser elaborado em nível de classe.
Construtor	O construtor é o método responsável pela inicialização de objetos de determinada classe no momento da criação dos mesmos. Este método não possui retorno.
Interface	O conceito de Interface permite que códigos sejam escritos de maneira independente da implementação. A Interface de uma classe é definida através de métodos públicos, uma vez que métodos privados não podem ser acessados de maneira externa.
Polimorfismo	O Polimorfismo prevê que uma operação pode ter seu comportamento realizado de forma diferente em classes distintas. Neste conceito também se enquadra a implementação de determinada operação por vários métodos, onde a linguagem de programação deve utilizar aquele que for mais adequado, de acordo com o nome, a classe e os argumentos. Este conceito engloba o recurso de Sobrecarga.
Abstração	A Abstração pode ser definida como o foco na tarefa que um objeto efetuará antes da sua implementação. Sendo assim, os detalhes são deixados de lado para que as características principais sejam destacadas.
Encapsulamento	O Encapsulamento consiste basicamente em "ocultar informações", permitindo uma filtragem de quais dados serão acessíveis por outros objetos, e quais não, por pertencerem ao funcionamento interno do mesmo. Este recurso é muito útil pois permite que um objeto seja modificado internamente sem comprometer outros objetos que o utilizam.
Persistência	A Persistência é a capacidade que um objeto possui de existir além da execução que o concebeu. É um recurso muito aplicado na manipulação das informações de um banco de dados por exemplo.

Tabela 1. Conceitos Importantes da Orientação a Objetos

Tipo/Método	Descrição
Class	Utilizado para iniciar a declaração de uma classe.
Constructor	O método construtor da classe, utilizado para a criação do objeto a partir de uma classe.
Destructor	O método utilizado para destruir um objeto.
Object	Esta declaração permite que determinada sub-rotina faça referência a um método de objeto.
Function	Uma rotina que retorna um valor.
Procedure	Uma rotina que não retorna um valor, salvo o fato de poder exportar informações através de parâmetros.
Inherited	Permite a chamada ao método presente na classe pai por um método da subclasse.
Virtual	Esta diretiva é aplicada em uma classe pai com o intuído de permitir a futura substituição de um método de mesmo nome presente em uma classe descendente.
Overload	Permite que duas ou mais sub-rotinas sejam criadas possuindo o mesmo nome.
Override	Permite que um método presente em uma subclasse substitua ou estenda o método virtual da superclasse.

Tabela 2. Conceitos Importantes presentes no Delphi para a Herança

chamado quando um objeto é criado, sendo assim, o construtor da classe pai é executado primeiramente e depois o da classe derivada (ordem normal). Já o destrutor tem seu funcionamento invertido, sendo executado o da classe derivada primeiramente, lembrando que este método tem a função de destruir o objeto. A manipulação desses métodos nas subclasses se justifica, pois, muitas vezes novos atributos, funções e procedimentos são adicionados à mesma, o que pode acarretar de acordo com a intenção, que tais elementos possam necessitar ser inicializados em conjunto com o objeto e, consequentemente, destruídos com ele.

Herança na prática em formulários

Para dar início à utilização de herança no Delphi, será criado um novo projeto File>New>VCL Forms Application. Um ponto que deve ser evidenciado é que este conceito de herança pode ser utilizado tanto na VCL quanto no novo ambiente FireMonkey do Delphi XE 2. Neste exemplo será criado um formulário com uma interface gráfica cujo propósito é fornecer a base para todos os formulários de cadastro do sistema. Sendo assim, para dar início, devem ser inseridos no mesmo todos os pontos em comum entre as telas de cadastro, tais como botões e outros componentes conforme desejado pelo desenvolvedor. Uma vez que o formulário padrão esteja finalizado, basta efetuar a gravação dele, onde neste projeto foi atribuído o nome de FormCadastros, bem como UFormCadastros para a Unit do mesmo. Para criar um novo formulário herdando o criado anteriormente de uma maneira extremamente simples, basta acessar o menu File>New>Other e, através da opção DelphiProjects>Inheritable Items selecionar o mesmo. A **Figura 1** apresenta a janela New Items do Delphi, uma das formas de aplicar Herança em Formulários de uma maneira rápida.

Entretanto, o Delphi provê outras formas para que a herança possa ser aplicada em um formulário por exemplo. Ao visualizar o código fonte de uma Unit, é possível notar facilmente que um formulário herda a classe TForm através do código FormX = class(TForm). Sendo assim, outra maneira de fazer com que um formulário herde outro, seria simplesmente alterar esta linha para a outra classe de formulário desejada. A **Listagem 1** apresenta a outra forma de aplicar herança com o formulário de cadastro pai à partir de outro formulário.

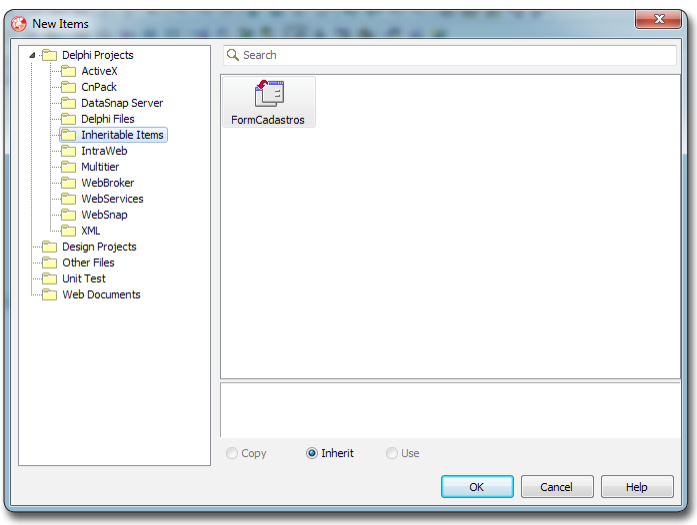


Figura 1. Janela New Items para a utilização de herança em formulários

Listagem 1. Aplicando Herança em formulário através do código fonte

```
unit UCadastroClientes;
...
uses
... , UFormCadastros;
type
 TFormCadastroClientes = class(TFormCadastros)
 private
  { Private declarations }
 public
  { Public declarations }
 end;

var
 FormCadastroClientes: TFormCadastroClientes;
```

Neste trecho de código é possível notar que a Unit UForm Cadastros foi declarada na primeira seção Uses e logo em seguida, o formulário atual, denominado TFormCadastroCliente herda a classe do formulário TFormCadastros através do código TFormCadastroCliente = class(TFormCadastros). Esta forma de herdar classes pode ser aplicada para outros componentes do Delphi também.

A grande vantagem que pode ser observada a primeiro momento é que o novo formulário herdou todas as características do formulário criado anteriormente, bem como seus objetos, sendo assim, vale a pena ressaltar também que as futuras alterações no formulário pai são automaticamente herdadas pelos seus filhos.

Aplicando herança e generalização em componentes

A aplicação de herança e generalização em classes e componentes do Delphi permite que o desenvolvedor possa criar seus próprios objetos baseados nos padrões existentes e principalmente alterar o comportamento dos componentes da própria ferramenta sem que tenha que recorrer ao uso de recursos de terceiros. A generalização é outro conceito da OO muito importante que pode ser definida como um relacionamento entre determinada classe e versões mais especializadas e refinadas da mesma. Uma situação muito comum que pode ser resolvida de uma maneira simples e que foi adotada neste artigo para exemplificar o uso de herança e generalização é mudar o foco de componentes como TEdits através das teclas ENTER, para avançar, e ESC, para retornar, em quantos formulários for desejado. Se não utilizar tais recursos, o desenvolvedor necessitaria percorrer cada formulário e determinar o funcionamento dos TEdits manualmente à sua maneira ou ainda defini-los um a um. Entretanto, neste exemplo convém criar uma Unit separada que será chamada por todos os formulários que possuem TEdits para que o funcionamento dos mesmos seja herdado através dela, bastando apenas adicioná-la na primeira seção Uses das Units de cada formulário a qual se deseja obter este efeito. Além de agilizar e facilitar o desenvolvimento, esta maneira de aplicar herança permite que, ao alterar a classe TEdit da Unit pai, as alterações ocorram para todos os TEdits associados. Para iniciar este exemplo basta inserir uma nova Unit através do File>New>Unit. Ela foi gravada com o nome de UAlteracaoComponentes em conjunto com o projeto. A **Listagem 2** apresenta a declaração do tipo TEdit para que o mesmo seja alterado e herdado pelos TEdits dos outros formulários.

Note que neste código inicial da UAlterarComponentes, após adicionadas as Units necessárias para a utilização dos recursos deste exemplo, a classe TEdit é herdada dela mesma para que possa ser estendida e tenha seu comportamento alterado de acordo com a nova necessidade. Também são criadas novas propriedades, FCor Anterior e FCorAnteriorTexto, para armazenar as cores de fundo e texto do objeto TEdit para que, quando o mesmo perder o foco, a cor antiga seja restaurada, uma vez que neste exemplo o TEdit também terá suas cores alteradas toda vez que receber o foco. Na sequência do código, alguns métodos são estendidos com o uso do override descrito anteriormente, sendo eles o construtor (constructor), executado no momento da criação do objeto, o DoEnter, executado quando o elemento recebe o foco, o DoExit, chamado quando o objeto perde o foco, e o KeyPress, para a manipulação de teclas pressionadas no mesmo. Também foi adicionado um novo método público para a classe TEdit (setFonte) com o intuito de demonstrar um simples exemplo de como aplicar o recurso de sobrecarga de métodos no Delphi através de overload. Para concluir, foram criadas duas constantes para armazenar as cores que serão utilizadas no momento em que o objeto ganhar o foco. Definido o tipo e as constantes, é o momento da implementação dos métodos. O primeiro método implementado foi o construtor. A **Listagem 3** apresenta o método construtor estendido.

Listagem 2. Criação da classe TEdit na Unit UAlteracaoComponentes

```
unit UAlteracaoComponentes;

interface

uses
  StdCtrls, Graphics, Forms, Messages, Classes;

type
 TEdit = class(StdCtrls.TEdit)
  private
   FCorAnterior: TColor;
   FCorAnteriorTexto: TColor;
  protected
   constructor Create(AOwner: TComponent); override;
   procedure DoEnter; override;
   procedure DoExit; override;
   procedure KeyPress(var Key: Char); override;
  public
   procedure setFonte; overload;
   procedure setFonte(Cor: TColor); overload;
 end;

const
 CorFocoFundo = $00CAFFFF;
 CorFocoTexto = $00000000; //Preto
```

Listagem 3. Método construtor estendido

```
constructor TEdit.Create(AOwner: TComponent);
begin
  inherited;
  CharCase := ecUpperCase;
end;
```

Note que neste trecho de código o comando inherited é utilizado para que o mesmo herde o método da classe pai e na sequência, a propriedade CharCase é definida como edUpperCase, fazendo assim com que todo o texto que seja digitado nos TEdits fique com letras maiúsculas. Neste exemplo fica evidente uma das utilidades de estender o método construtor, ou seja, no momento em que o objeto é criado através do mesmo, as propriedades padrões dele podem ser alteradas por outra desejada.

O próximo método que será estendido é o DoEnter. A **Listagem 4** apresenta o código atribuído a este método.

É possível observar neste trecho de código que após a chamada ao método da classe pai, as cores atuais de fundo e fonte

do TEdit são armazenadas nas propriedades FCorAnterior e FCorAnteriorTexto sucessivamente. Na sequência, as novas cores são atribuídas ao objeto de acordo com as constates definidas anteriormente, através das propriedades Color, para a cor de fundo, e Font.Color, para a cor da fonte, lembrando que o método DoEnter é executado toda vez que o objeto ganhar o foco.

Já o método DoExit é executado no momento em que o componente perde o foco, e ele também será estendido. A **Listagem 5** demonstra o código que foi aplicado neste método.

Listagem 4. Método DoEnter estendido

```
procedure TEdit.DoEnter;
begin
  inherited;
  FCorAnterior := Color;
  FCorAnteriorTexto := Font.Color;
  Color  := CorFocoFundo;
  Font.Color := CorFocoTexto;
end;
```

Listagem 5. Método DoExit estendido

```
procedure TEdit.DoExit;
begin
  inherited;
  Color := FCorAnterior;
  Font.Color := FCorAnteriorTexto;
end;
```

Após a chamada do método da classe pai, a propriedade Color (cor de fundo) do objeto recebe a cor de fundo antiga, FCorAnterior, e a propriedade Font.Color (cor da fonte) recebe a cor de fonte antiga, FCorAnteriorTexto.

Como pôde ser observado, ao estender os métodos DoEnter e DoExit, existe a possibilidade de se manipular as ações que serão realizadas sempre que o componente ganhar ou perder o foco, podendo ser empregadas diversas utilidades neste momento.

O último método que será estendido é o KeyPress, que por sua vez é executado toda vez que uma tecla for pressionada sobre o objeto, permitindo assim definir funções padrões para as teclas, como é o caso de mandar o foco para o próximo componente. A **Listagem 6** apresenta o código que foi utilizado para estender o método KeyPress do TEdit juntamente com o parâmetro Key, o valor da tecla que foi pressionada.

Neste trecho, após a chamada do comando inherited, é efetuado um teste com o intuito de verificar se o objeto TEdit em questão já possui métodos para controle de teclas definidos, ou seja, se o desenvolvedor desejar anular o efeito deste método em apenas um TEdit mas utilizá-lo em outros presentes no mesmo formulário, basta ele preencher um dos eventos para a manipulação de teclas, OnKeyPress, OnKeyDown ou OnKeyUp do TEdit para que este método não seja mais executado sobre o mesmo. Isto é possível através do uso da função Assign(). Este comando é muito útil e retorna o valor verdadeiro (true) caso o valor do parâmetro especificado, que pode ser um método, ponteiro ou objeto, não seja nulo (igual a nil), evidentemente retornando falso caso contrário. Sendo assim se não foram definidos os eventos testados, o método estendido é adotado com o intuito de que, no momento em que o usuário pressione a tecla ENTER (valor #13), o foco seja direcionado para o próximo componente, e a tecla ESC (valor #27), o foco volte para o componente anterior ao mesmo de acordo com o Tab Order. O trecho que contém o valor Key := #0 tem o intuito neste exemplo de anular o áudio que é produzido com a utilização da tecla Enter. Observe que é efetuada uma chamada ao Screen.ActiveForm, com a finalidade de obter o formulário ativo atual, em conjunto com o Perform(), responsável por enviar uma mensagem para o Windows (Windows Messages), sendo neste caso a mensagem WM_NEXTDLGCTL, utilizada para alterar o foco de objetos.

Listagem 6. Aplicando herança em formulário através do código

```
procedure TEdit.KeyPress(var Key: Char);
begin
  inherited;
  if not Assigned(TEdit(Self).OnKeyPress)
    and not Assigned(TEdit(Self).OnKeyDown)
    and not Assigned(TEdit(Self).OnKeyUp) then
  begin
    if (Key = #13) then
    begin
      Key := #0;
      Screen.ActiveForm.Perform(WM_NEXTDLGCTL,0,0);
    end
    else if (Key = #27) then
    begin
      Key := #0;
      Screen.ActiveForm.Perform(WM_NEXTDLGCTL,1,0);
    end;
  end;
end;
```

Listagem 7. Sobrecarga do método setFonte

```
procedure TEdit.setFonte;
begin
  Self.Font.Style := [];
  Self.Font.Color := CorFocoTexto;
end;

procedure TEdit.setFonte(Cor: TColor);
begin
  Self.Font.Style := [fsBold, fsItalic];
  Self.Font.Color := Cor;
end;
```

Com os métodos estendidos, o próximo passo deste exemplo é demonstrar uma forma simples de aplicar o conceito de sobrecarga de métodos. A **Listagem 7** apresenta um exemplo de sobrecarga de métodos para a definição da aparência de um objeto TEdit.

É possível notar que a diferença entre ambos os procedimentos do método sobrecarregado setFonte() esta na relação de seus parâmetros, onde o primeiro não possui nenhum e o segundo possui um,

cabendo ao Delphi selecionar automaticamente o mais adequado de acordo com os parâmetros especificados durante uma futura chamada do mesmo. Neste exemplo, o primeiro método é utilizado para limpar uma formatação que possa existir na Fonte do TEdit, tal como negrito, itálico e cor. Já o segundo aplica o estilo negrito (fsBold) e itálico (fsItalic) na fonte do TEdit, alterando também a cor de acordo com o parâmetro que foi fornecido.

Nota do DevMan

A classe Screen é muito útil para ser aplicada sempre que se deseja obter uma informação do estado da tela atual em que o aplicativo esta sendo executado, como a resolução, bem como obter o formulário atual e componente ativo da aplicação. Sendo assim, também é possível enviar mensagens para o Windows de acordo com o objeto que está ativo por exemplo.

Nota do DevMan

O Windows Message (Mensagem Windows) é um recurso presente no ambiente Windows cujo objetivo é efetuar a troca de mensagens entre uma aplicação e o próprio Sistema Operacional. As mensagens são informações que vão desde simples ações, como um clique do mouse ou maximização de uma janela, até recursos mais complexos. No Delphi, isto acontece de uma forma transparente para o desenvolvedor pois ele abstrai muitas destas mensagens através de seus eventos, facilitando assim o trabalho no desenvolvimento das aplicações, mas o desenvolvedor pode interceptar ou enviar tais mensagens manualmente sempre que desejar ou necessitar, através de métodos como Perform() e SendMessage() por exemplo, podendo também criar procedimentos em casos onde determinado evento não esta implementado para um objeto por padrão. No entanto, para manipular ou enviar as mensagens, existe a necessidade de se conhecer as definições das mesmas a fundo, ou seja, para selecionar um próximo componente por exemplo, a mensagem WM_NEXTDLGCTL deve ser enviada e assim por diante.

Nota do DevMan

O Tab Order (Ordem de Tabulação) consiste na determinação de uma sequência para a atribuição do foco em componentes permitidos de um formulário, selecionando assim o próximo quando o usuário utiliza a tecla TAB e o anterior através da combinação SHIFT + TAB por padrão. O Tab Order de um formulário pode ser facilmente acessado através do menu Edit>Tab Order ou de um clique com o botão direito sobre um formulário por exemplo.

Uma forma de aplicar ambos os métodos em tempo de execução para visualizar um simples efeito deste recurso foi através do pressionamento de teclas. Neste exemplo, ao pressionar a tecla F2, o primeiro método é executado, e F3, o segundo é chamado, através do evento OnKeyUp do formulário, que é acionado quando uma tecla é solta. Sendo assim, o formulário TFormCadastroCliente foi utilizado, onde foram adicionados 5 TLabels e 3 TEdits. A **Figura 2** apresenta o visual final que este formulário terá.

Para o funcionamento desejado, a propriedade KeyPreview do formulário deve ser definida como True, possibilitando assim que o mesmo capture a tecla pressionada. A **Listagem 8** apresenta um código de exemplo que foi adicionado no evento OnKeyUp do formulário que possibilita a chamada a ambos os métodos de acordo com a tecla pressionada pelo usuário.

Verificamos se a tecla pressionada é igual ao F2. Satisfeita a condição, um novo teste é realizado para identificar se a classe do controle ativo é igual a TEdit através de ActiveControl.ClassType = TEdit, e caso seja verdadeiro, então o objeto do tipo TEdit ativo chama o método setFonte através de TEdit(ActiveControl).setFonte. No próximo item ocorrem os mesmos testes porém com a tecla F3 e neste caso o método é chamado com a passagem do parâmetro clRed, a cor vermelha.

Como pode ser notado, tanto o recurso de estender métodos quanto o de sobrecarga unidos a generalização da classe TEdit, permitem que o mesmo possa ter seu comportamento desempenhado de acordo com o desejo do desenvolvedor, utilizando os próprios recursos do Delphi, sem que exista a necessidade da adição de componentes adicionais.

Figura 2. Exemplo do visual do formulário TFormCadastroCliente

Listagem 8. Evento OnKeyUp adotado como exemplo para a utilização do método setFonte

```
procedure TFormCadastroCliente.FormKeyUp(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  inherited;
  if Key = VK_F2 then
    if ActiveControl.ClassType = TEdit then
      TEdit(ActiveControl).setFonte;
  if Key = VK_F3 then
    if ActiveControl.ClassType = TEdit then
      TEdit(ActiveControl).setFonte(clRed);
end;
```

Conclusão

A Orientação a Objetos é um conceito que traz consigo uma diversidade de recursos que englobam inúmeras vantagens para seus utilizadores. Dentre tais recursos, um que merece grande destaque é a herança, uma das bases da OO e que permite um aumento da produtividade tanto no desenvolvimento quanto na manutenção de um software. Ela possibilita que simples classes possam ser expandidas e especializadas através da generalização, sendo as mesmas transformadas em classes complexas conforme vão surgindo novos descendentes.

No Delphi, a herança já é aplicada de forma natural quando um formulário ou outro componente é adicionado ao projeto. Entretanto, é possível utilizar recursos da própria ferramenta para explorar mais afundo a aplicação de herança e fazer assim com que as classes e objetos funcionem da maneira mais adequada para o

projeto, sem que exista a necessidade de repetir o mesmo processo para cada componente descendente de uma mesma classe.

Este artigo teve o intuito de apresentar de uma maneira bem simples, uma introdução a algumas formas de aplicar herança e alguns outros recursos em formulários e classes através do Delphi, demonstrando suas vantagens, sugerindo assim uma alternativa para ser adotada sempre que situações similares às apresentadas surgirem, mas vale à pena evidenciar que estes recursos da OO vão muito além daquilo que foi comentado e evidentemente exigem um maior aprofundamento para possibilitar o desenvolvimento de aplicações cada vez mais refinadas.

DevSpace do Autor
http://www.devmedia.com.br/space.asp?id=283274

Giuliano Scombatti Pinto
giuliano@sygnux.com.br
É Analista Desenvolvedor da Sygnux Software (www.sygnux.com.br) localizada em Monte Alto/SP, atuando também com desenvolvimento Web e Android. Formado pela Fatec/TQ, começou a desenvolver com Delphi em 2003. Atualmente trabalha com Delphi, Java, PHP e Flash/Flex. Professor de Informática de Projetos Sociais da Prefeitura Municipal de Monte Alto/SP.

Dê seu feedback sobre esta edição!

A Clubedelphi tem que ser feita ao seu gosto. Para isso, precisamos saber o que você, leitor, acha da revista!

Dê seu voto sobre este artigo, através do link:

www.devmedia.com.br/clubedelphi/feedback

DÊ UM SALTO
EM CONHECIMENTO!

Acesse o maior portal para desenvolvedores da América Latina!

17 mil posts

280 mil cadastrados

5 milhões de page-views por mês

http://www.devmedia.com.br/

DevMedia

ITIL, além do desenvolvimento

Gerencie sua empresa de TI

Paulo Quicoli

A grande maioria dos desenvolvedores são pessoas extremamente técnicas. São pessoas que sempre estão dispostas a criar algo e com isso sempre pensam em ter seu próprio negócio, isso se ainda não o tem. Quando conseguem seu próprio negócio percebem que é mais difícil administrar do que programar. Mas toda essa situação se agrava quando essa empresa, gerida por alguém técnico, começa a crescer e seu gestor precisa deixar totalmente o trabalho de desenvolvimento. Agora sua obrigação é administrar uma empresa de TI.

Existem técnicas e metodologias que podem auxiliar essa gestão, dando mais clareza aos envolvidos. Neste artigo trataremos do ITIL e como ele auxilia na gestão dos serviços que uma empresa de TI pode possuir. É isso mesmo, uma empresa de TI não é formada apenas por desenvolvimento, mas por serviços.

ITIL e sua história

Esta sigla significa biblioteca de infraestrutura de tecnologia da informação. Sua idealização aconteceu nos anos 80 no Reino Unido através de seu órgão governamental responsável pelo setor de TI da época. Este órgão era responsável por administrar a prestação de serviço que o governo recebia de terceiros. Isso gerou informação suficiente que o fez identificar um possível caos à sua frente. Foi questionada a falta de padronização na gerência da TI por parte geral, tanto interna do governo quanto desses terceiros, o que resultava em desperdício financeiro. Assim, para tentar organizar e otimizar foram desenvolvidas recomendações, as chamadas boas práticas de gerenciamento de serviços de TI.

Essas boas práticas foram compiladas em livros – daí o nome biblioteca – sendo que em sua primeira versão continha trinta e um livros, abordando desde o cabeamento de redes até um planejamento de contingência de sistemas de TI. Em 2000 foi elaborada sua segunda versão, que agora possuía apenas sete livros, que foram reorganizados. Em seu conteúdo abordava-se orientações sobre processos e melhoria contínua, além claro do conteúdo original revisto. Todo este trabalho fez com que a versão dois fosse reconhecida mundialmente como um padrão, que em anos seguintes se tornaria uma normativa. O Instituto de Padronização Britânico criou em 2002 a norma BS 1 5000 pautada no ITIL e em 2005 o ISSO criou a norma ISSO/IEC 20000, baseando-se na norma britânica.

Em 2008 houve uma terceira atualização do ITIL, chegando à sua versão três, reorganizando a versão anterior de sete, para cinco livros. Contudo seu conteúdo não é menor, mas maior.

Resumo DevMan

De que trata o artigo:

O artigo trata da metodologia ITIL, que ajuda as empresas a organizar e padronizar a qualidade do serviço da equipe de TI.

Em que situação o tema é útil:

Para empreendedores que originalmente eram técnicos e que desejam saber como organizar suas empresas com um padrão internacional de qualidade, preparando-as para o crescimento organizado e constante.

ITIL, além do desenvolvimento.:

Neste artigo tratamos de algo que não está diretamente relacionado a desenvolvimento de software, mas está ligado com a organização da empresa que produz softwares, ou melhor, a qualquer empresa. Se você é empreendedor, mas ainda se envolve com a parte técnica, talvez não tenha tido tempo de se envolver ou até mesmo conhecer metodologias organizacionais que tem por objetivo preparar sua empresa para um crescimento. Todos querem crescer, mas é preciso crescer de forma organizada. Vamos explorar a metodologia ITIL, mostrar uma visão geral de seu ponto de vista sobre serviços e ver como isso pode se encaixar em algumas situações.

Sua necessidade

Os negócios de uma forma geral estão cada vez mais dependentes da TI, assim, seus gestores buscam formas para padronizar e garantir uma qualidade de serviço, seja fornecido ou recebido. Nessa qualidade entenda-se também redução de custos e agilidade nos processos. Nisso o ITIL tem ganhado destaque por ser totalmente independente de plataforma ou tecnologia, possuindo atividades que podem se encaixar em empresas de qualquer porte. Essas atividades fazem parte de um processo que cobre uma ou mais áreas da TI, como desenvolvimento de serviços, gerenciamento de infraestrutura e prestação de suporte. Assim, seu objetivo é a descrição clara e eficiente que garante qualidade aos clientes.

Implantar a metodologia ITIL pode trazer os seguintes ganhos/benefícios: ganho de tempo por já ser uma prática atestada por outros, geração de processos mais eficientes que podem trazer agilidade aos resultados, aumento de qualidade de serviços, alinhamento dos serviços de TI com as necessidades do negócio envolvido na empresa, aumento de satisfação de clientes, visão mais clara da capacidade atual e das principais falhas.

Definições básicas

O ITIL possui algumas definições conceituais que precisam ficar claras para que continuemos a entender o que o ITIL quer resolver. A palavra serviço é um termo que é definido como uma forma de entregar valor para seus clientes, mas de uma forma que esse valor atinja os resultados esperados de forma fácil. Isso pode ser obtido quando esse serviço ou produto atende especificações que foram determinadas e acordadas com seu cliente. Outra ideia a ser definida é o gerenciamento de serviços. Este se trata de um conjunto de rotinas/habilidades organizacionais para gerar valor na forma de serviços. Esse valor pode ser obtido através da união de duas expectativas: utilidade + garantia = valor.

A utilidade nada mais é que a funcionalidade do serviço, o que ele faz. Isso se inclui suas características e objetivos. Já a garantia envolve a capacidade, desempenho, disponibilidade e a adequação ao uso. Tudo isso gera o chamado valor para o cliente. Por valor podemos entender como aquilo que conquista o cliente/usuário. Imaginando um sistema informatizado de controle de vendas, o cliente espera que esse sistema venha suprir suas necessidades que foram previamente descritas durante uma fase de análise e que uma vez implantado, funcione como especificado, mas além disso, ele espera que quando precisar de um suporte tenha pessoas que falem a "sua língua". Ou seja, ele precisa também não só de um sistema especializado, mas de um suporte especializado em suas necessidades.

Para que uma empresa em crescimento possa gerar esse valor ao seu cliente, ela precisa ter uma definição bem clara sobre funções, papéis, processo e responsabilidades. Funções são equipes e ferramentas utilizadas para realizar um processo ou trabalho. Por exemplo o suporte. Papéis são as responsabilidades que devem ser definidas às pessoas que participam em um processo. É muito importante para as pessoas saberem exatamente qual o seu papel dentro de uma organização, dentro das rotinas de seu ambiente de trabalho. Como exemplo podemos citar uma pessoa que trabalha como help desk. Esta pessoa sabendo claramente seu papel, sabe até onde se envolver. Processos são uma sequência coordenada de atividades, que são executadas por regras e tem o objetivo de gerar valor ao cliente. Mais uma vez cito o setor de suporte de uma empresa. O processo de atendimento do setor de suporte deve ter sido desenhado de tal forma que o resultado seja satisfação, e não mais reclamação. Por fim temos o conceito de responsabilidades. Este é o compromisso em responder por atos ou por algo que foi confiado. Para se alcançar o valor, é preciso pessoas responsáveis.

A vida de um serviço

O ITIL foca em algo chamado de ciclo de vida do serviço. Um serviço passa por fases para que possam atender às necessidades estratégicas da empresa e acompanhar as mudanças constantes, seja de requisitos ou de negócios. Essas fases são de estratégia, desenho, transição, operação e melhoria continuada. Cada fase é compreendia por uma série de processos, vamos abordar essas fases.

Fase de estratégia

Ao crescer, uma empresa precisa pensar em seu futuro e não vai ser levada de um lado para o outro, simplesmente reagindo às situações ou ao mercado. Ela precisa definir onde quer estar, como estar, quando estar. Essas são metas, objetivos que devem ser específicos, mensuráveis, atingíveis, relevantes e temporais. Isso exige uma participação da alta direção, com comprometimento, pois o desejo de crescimento deve manter os serviços já existentes com qualidade e, introduzir novos para o aumento dessa qualidade. É na fase estratégica que uma empresa faz a reflexão sobre seu futuro e utiliza isso como sinergia para todos seus colaboradores. Nesta fase também são definidas as visões de médio e longo prazo, isso inclui analisar o mercado atuante, as necessidades de clientes, além de identificar o que a empresa tem de melhor e de pior no momento. Com todas essas informações na mão é traçado um plano de ação. Este envolve todos que participam da empresa. Ao distribuir o plano de ação através das tarefas, é necessário evidenciar o papel dos envolvidos. Existe um documento, na verdade uma matriz, chamado RACI. É uma lista separada por atividades que indica a posição das pessoas como a seguir:

R – Responsible. Responsável por realizar a atividade, devem reportar ao dono da mesma;

A – Accountable. É quem aprova ou não a atividade, é o dono da mesma, que toma a decisão final;

C – Consulted. É aquela pessoa que precisa ser consultada para a realizada da atividade;

I – Informed. Pessoa que precisa ser informada do término da tarefa, talvez seja ninguém.

A fase de estratégia envolve algumas atividades de gerenciamento que podem conflitar com a cultura existente em empresas que não praticam o ITIL. Essas empresas normalmente possuem uma cultura chamada de "urgentismo". Elas acreditam que tudo deve acontecer de forma mais rápida possível e que processos são mera burocracia.

As atividades envolvidas neste caso são: gerenciamento de portfólio, gerenciamento de demanda e gerenciamento financeiro.

Portfólio

A gerência de portfólio diz respeito a como que se administra, o quanto e como se investe em um serviço. Além disso ela também deve monitorar se o mesmo agrega valor ao negócio. É composta por três subconjuntos: funil de serviços (serviços pedidos e que estão em desenvolvimento), catálogo de serviços (serviços em produção e que estão em disponíveis para uso) e serviços obsoletos (serviços que não estão em produção mais).

Antes de um serviço ser coloca em produção ele passa por uma análise, que é o funil. Nesse momento é que se encontram serviços que não estão atendendo mais, que não dão retorno e consequentemente precisam ser substituídos.

Demanda

É responsável por identificar as necessidades de recursos de TI para acompanhar a evolução do negócio. É preciso identificar demandas atuais e futuras e em quais períodos que o cliente utiliza mais um determinado serviço e como. Para auxiliar nesse levantamento alguns conceitos sobre a empresa precisam ser identificados:

- **Perfil de usuário**: é o padrão de um usuário para determinado perfil. A empresa precisa entender seu público;
- **Serviço principal**: é o serviço que atende às necessidades essenciais de um ou mais clientes;
- **Serviço de apoio**: é o que sustenta ou provê melhorias em um serviço principal. Por exemplo, sua empresa desenvolve um sistema ERP, este é seu serviço principal, contudo, nele existe uma ferramenta de backup. Esse backup é um serviço de apoio;
- **Pacote de serviço**: É o detalhamento da utilidade e garantia, visando um nicho de mercado específico ou perfil de usuário.

Financeiro

Toda decisão estratégica deveria ser orientada por informações financeiras. Nem sempre uma empresa que está crescendo possui essas informações. Contudo estas são importantes para quantificar os valores financeiros dos serviços e assim obter o ROI (return of investimento) que nada mais é que o retorno de investimento. A gerência do processo financeiro é constituída por três subprocessos:

- **Orçamento**: é o planejamento do quanto se investir;
- **Contabilidade**: identificação dos custos, que poderão ser por serviço, cliente, por atividade, etc;
- **Cobrança**: é utilizado para recuperar os recursos investidos.

Desenho

Depois de toda a análise anterior e coleta dessas informações, chega o momento de criar os serviços baseando-se no que foi levantado até o momento. O desenho de um serviço envolve cinco fases distintas: requisitos e solução, portfólio, tecnologia e arquitetura, processos e métricas. A primeira delas está ligada ao levantamento de requisitos. É preciso analisar cada requisito e o custo da solução. O portfólio por sua vez mostra como os serviços oferecidos agregam valor. Tecnologia e arquitetura dizem respeito à infraestrutura, seus relacionamentos e o ambiente em que eles pertencem. Ela deve mostrar o relacionamento de todos os serviços que suportam um negócio, esboços de aplicações, como informações são gerenciadas e divulgadas, como está distribuída a infraestrutura no negócio e como é o ambiente físico. O quarto item, processos, especifica que eles devem ser desenhados e que se relacionem entre si. Por fim temos as métricas que podem ser utilizadas para medir a eficácia, eficiência, progresso e aderência.

Espera-se que ao fim desta fase um pacote de desenhos de serviço para direcionar a transição de serviço, ou seja, serão identificadas todas as informações para colocar um serviço em operação.

Fase de transição

Esta fase tem a função de preparar o serviço e colocá-lo em produção de forma que cause o menor impacto possível, controlando os riscos de falhas e interrupções. Ela também é utilizada para medir a satisfação do cliente no negócio, ou seja, é preciso ir ao cliente acompanhar para ter certeza de que tudo o que foi entregue foi o que ele pediu. Para aplicar esta fase as seguintes gerências são necessárias: planejamento e suporte de transição, gerenciamento de liberação e implantação, validação e teste de serviço, gerenciamento de avaliação, gerenciamento de mudança, gerenciamento da configuração e de ativos de serviços e gerenciamento do conhecimento. Vamos abordar cada uma de forma simplificada.

- **Planejamento e suporte da transição** – O objetivo é planejar e coordenar recursos e pessoas para estabelecer um novo serviço. Isto exige tarefas como gerar uma visão geral para definir como será a abordagem para a transição, recolhimento de toda documentação disponível e indispensável para início da mudança, identificar as atividades necessárias para a execução e liberação em ambiente de produção. Se tudo for feito de forma eficaz, será muito mais fácil lidar com grandes conjuntos de mudanças.
- **Gerenciamento de liberação e implantação** – Aqui os serviços são construídos e testados de acordo com o que foi definido no desenho do serviço. É aqui também que deve ser garantida a transferência de conhecimento, assim os usuários poderão tirar o máximo de proveito. Ao aplicar eficientemente esse gerenciamento, os seguintes benefícios são esperados: aumento da produtividade dos usuários, redução na quantidade de mudanças drásticas, participação dos usuários nos testes de liberações, estabilização do ambiente e outros mais. Em se tratando de uma empresa de software, podemos relacionar este gerenciamento a uma implantação de software ou atualização do mesmo. Existem alguns passos que podem ser tomados para auxiliar nos resultados, como a realização de testes e pilotos, onde se faz a homologação junto ao cliente, planejamento e preparação para a implantação em si, uma verificação da mesma para realizar uma revisão geral e então o fechamento, quando é feita a última revisão e encerra-se a implantação. Depois de realizada, deve-se colher alguns indicadores de desempenho, como o número de incidentes causados pela liberação, percentual de incidentes na fase de construção e de testes de liberação, percentual de liberações feiras dentro do tempo e com recursos planejados, evidências de reversões não autorizadas de versões.
- **Validação e teste de serviço** – Os testes são essenciais e devem ser feitos com seriedade, pois se algo não testado for colocado em produção, aumentará o número de incidentes, erros serão mais difíceis de diagnosticar e o custo de correção será alto. Mas o pior é a insatisfação do cliente por não receber algo com o valor desejado. Para aplicar essa validação é necessário realizar algumas atividades: o planejamento, onde se criam as etapas, tempo de entrega, aceitação, enfim, todo recurso necessário para a execução dos testes; plano de testes, usado para garantir

que todos os procedimentos de execução dos testes sejam documentos, precisos e claros; Preparação do ambiente, cria-se o ambiente para o projeto piloto ou testes; Teste, realiza os testes de acordo com o plano de testes feito; avaliação, compara-se os resultados obtidos com os esperados.

• **Avaliação** - Neste processo é avaliada a performance do produto, se é aceitável ou não.

• **Gerenciamento de mudança** – É administrar as mudanças que podem ocorrem, ou seja, elas devem ser registradas, avaliadas, autorizadas, priorizadas, planejadas, testadas, implementadas, documentadas e revisadas. Nesse processo temos algumas tarefas: conselho consultivo de mudanças, é um grupo ou uma pessoa que autoriza e avalia as mudanças. Geralmente é o detentor da regra de negócio ou conhecedor técnico maior, dependendo do tipo de mudança; Requisição de mudança, é o pedido formal para realizar a mudança, esse documento pode ser eletrônico ou manual, mas deve conter o detalhamento do que é preciso mudar; Programação de mudança, é uma lista que contém todas as mudanças aprovadas e seu cronograma de implantação. Uma mudança pode ser do tipo normal, que é feita por meio da requisição; padrão, que é algo já acordado entre as partes, algo rotineiro e de baixo impacto; e emergencial, que são relativas a falhas encontradas. Para o bom gerenciamento de mudanças, a cada nova solicitação de mudança as seguintes perguntas devem ser respondidas: Quem requisitou a mudança? Qual a razão? Qual o retorno esperado? Quais os riscos? Quais recursos necessários? Quem é o responsável pela construção, testes e implantação? Qual o relacionamento com outras mudanças?

• **Gerenciamento da configuração e ativos de serviços** – Este serve de apoio aos demais processos, pois disponibiliza informações exatas da máquina dos usuários, além de saber ser há algum gasto desnecessário ou se existem itens não autorizados na rede.

• **Gerenciamento do conhecimento** – As informações devem estar disponíveis durante o ciclo do serviço, mas essas devem ser confiáveis e entregues no local adequado ou pessoa competente. Com isso pode-se assegurar que os envolvidos tenham uma compreensão clara e igual do valor a entregar.

Figura 1. Ciclo PDCA

Fase de operação de serviço

Esta fase cuida do dia a dia do serviço e é aqui que o cliente verifica se está recebendo o valor esperado. Para garantir isso é preciso realizar uma administração do serviço prestado, como monitorar desempenho e lidar com conflitos de forma equilibrada. Esse equilíbrio muitas vezes diz respeito ao que é desejado pelo cliente e o que deve ser realmente implementado. O desenvolvimento deve acompanhar as regras do negócio envolvido sempre. Se isto não for pensando, acaba-se prometendo algo ao cliente que não pode ser entregue.

Essa administração também inclui um gerenciamento de incidentes. Esta atividade tem por objetivo restaurar a operação normal do que foi acordado com o cliente, de forma mais rápida possível. Se você ofereceu um sistema que possui o serviço de imprimir em impressoras térmicas e por algum motivo ele não está fazendo isso, este incidente deve ser administrado. O gerenciamento é o primeiro atendimento de falhas, assim, é o que o cliente mais vê. Este processo é acompanhado por indicadores-chaves de desempenho que são coletados pelo ciclo de vida do incidente, e com base nessas informações é realizado um levantamento que indica se o tratamento de incidentes está eficaz ou não. Isso gera um avalição e plano de ação para correção, ou seja, estamos aplicando aqui o conhecido PDCA, **Figura 1**.

Existe também a gerência de problemas, que comumente confundida com a gerência de incidentes. É preciso entender que um incidente não se transforma em problema, mas, ele é a porta de entrada de um problema. Um incidente gera a

abertura de um registro de problema. Se lermos novamente o objetivo da gerência de incidentes, veremos que sua motivação é reestabelecer o serviço, já para os problemas a motivação é outra. É preciso encontrar a causa raiz do problema e eliminá-la. Uma vez encontrada a causa, esse problema passa a ser conhecido, bem como sua correção. Falando agora especificamente de sistema, outro ponto que pode passar despercebido por quem está crescendo é a gerência de acesso. Sua finalidade é garantir que apenas usuários autorizados possam acessar os serviços disponíveis. Alguns termos dessa gerência precisam ficar claros:

• Acesso – Nível de funcionalidade de um serviço;

• Identidade – É a identificação do usuário, é o que torna único dentro da organização;

• Direito/Permissão – Identifica quais acessos o usuário possui;

• Serviços de diretório – Banco de dados que gerencia os direitos e permissões dos usuários.

Muitas vezes um sistema possui um controle de usuário, mas também muitas vezes o usuário não é esclarecido da importância da identidade. Com ela é possível encontrar qual usuário que realiza uma dada operação e gerenciar operações de risco. Se isso não for esclarecido, é bem provável que o usuário crie apenas um usuário, ou um por setor e coisas do tipo. Além dessa conscientização, essa gerência deve ter as seguintes atividades: requisitar acesso, ou seja, deve solicitar a identidade para execução de um serviço; verificar o acesso, parece óbvio mas o sistema deve verificar se a identidade informada corresponde; Conceder direitos, através de alguma política do cliente deve-se permitir a concessão e revogação de direitos dos usuários cadastrados; Monitorar status das identidades, esse cadastro de usuários deve estar sempre atualizado, se um determinado usuário não pertence mais ao setor ou empresa, sua conta no sistema deve ser atualizada ou excluída; fazer registro e acompanhar o acesso, uma vez logado todas as ações devem ser registradas para futura verificação.

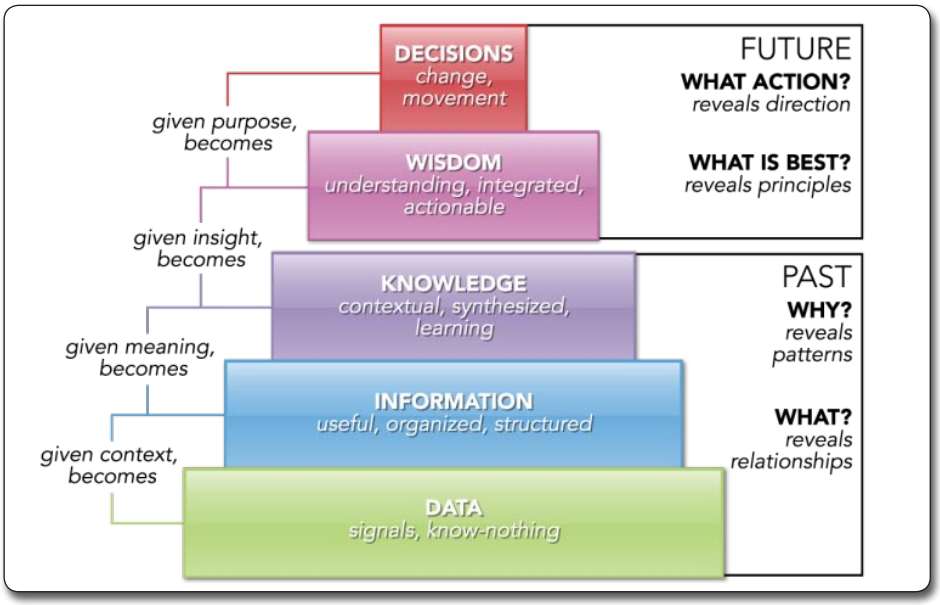


Figura 2. Modelo DIKW

Com este controle, muitos incidentes podem ser evitados.

Fase de melhoria continuada

Aqui é examinado, analisado e identificado melhorias para todo o processo. Um processo bom é um processo que é revisto sempre. Um processo não é feito para ficar sempre como foi inicialmente planejado. Ao término de cada fase, é gerada informação que pode ser utilizada para a fase seguinte, tornando isso um ciclo contínuo. A fase estratégica tem a visão do negócio, na fase do desenho do serviço é projetado o que o negócio precisa, na transição esse desenho é passado para o operacional e lá esse serviço é mantido para os clientes. A melhoria continuada analisa cada fase, ela não está posicionada no fim da cadeia dos processos, mas sim, durante cada processo. É sua execução durante as fases que irá garantir a eficiência do serviço como um todo. Para garantir esse objetivo alguns modelos são utilizados. A começar pelo PDCA já comentado aqui. Temos também o CSI model (continual service improvement). Este é composto por uma série de perguntas que tem a finalidade de responder uma pergunta principal: Como mantemos o que conquistamos? A série de perguntas são estas a seguir:

• Qual é nossa visão? – É preciso ficar claro para todos qual a visão do negócio que a empresa possui. Isso envolve a definição de uma missão, objetivo e metas claras;

• Onde estamos agora? – A posição atual da empresa junto ao mercado, junto aos seus clientes. Esse é um levantamento que identifica a empresa para todos;

• Onde queremos estar? – É a criação de metas mensuráveis, priorizando cada uma delas conforme a necessidade dos clientes e também dos objetivos da empresa;

• Como chegaremos lá? – Ficando claro onde se deseja estar, então se traça um plano de melhoria de serviços e processos;

• Já chegamos lá? – Não basta simplesmente seguir os processos, é preciso avaliar os resultados e identificar se o lugar almejado já foi alcançado;

Mas onde está a resposta para a primeira pergunta? A resposta está em manter este ciclo de auto conhecimento, auto análise da empresa e seus serviços de forma constante. Isso faz com que não se perca o foco do objetivo. E para ajudar a responder essas questões existe um conceito interessante no modelo de DIKW (Data, Information, Knowledge, Wisdom): Os dados capturados geram informação, a informação gera conhecimento que por sua vez gera sabedoria e esta reflete por todos, gerando ações (**Figura 2**).

Conclusão

Ao enxergar sua empresa como um todo, identificando seus serviços e valores é natural realizar os questionamentos sugeridos pelo modelo CSI. O ITIL está aí para colaborar com a organização de sua empresa e alcance de geração de valor para seu cliente, com isso sua empresa pode destacar-se de seus concorrentes. Claro que, se você ainda é pequeno talvez não seja interessante procurar por um profissional certificado nessa metodologia organizacional, mas é importante que se assimile os conceitos apresentados e se tente aplicar algo disso tudo. Às vezes você já faz alguma coisa disto e não sabia.

Neste artigo vimos apenas uma casca do que pode ser feito, porém a essência do todo foi apresentada. Se observarmos tudo o que foi apresentado pode-se resumir como um plano. Pensa-se na identificação do que fazer, o quanto fazer, para quem fazer e quem ficará responsável. Então se pensa no custo de tudo isso, o que é preciso investir. Então se constrói a ideia e esta é preparada para entrar em produção. Em produção ela é monitorada constantemente para saber se está no nível esperado. E uma vez estando, se pensa no futuro novamente, aplicando a autoanálise. Este é o ciclo básico e essencial. Espero que este artigo tenha despertado todos os empreendedores para o assunto, e assim que todos melhorem e criem suas empresas prontas para crescer.

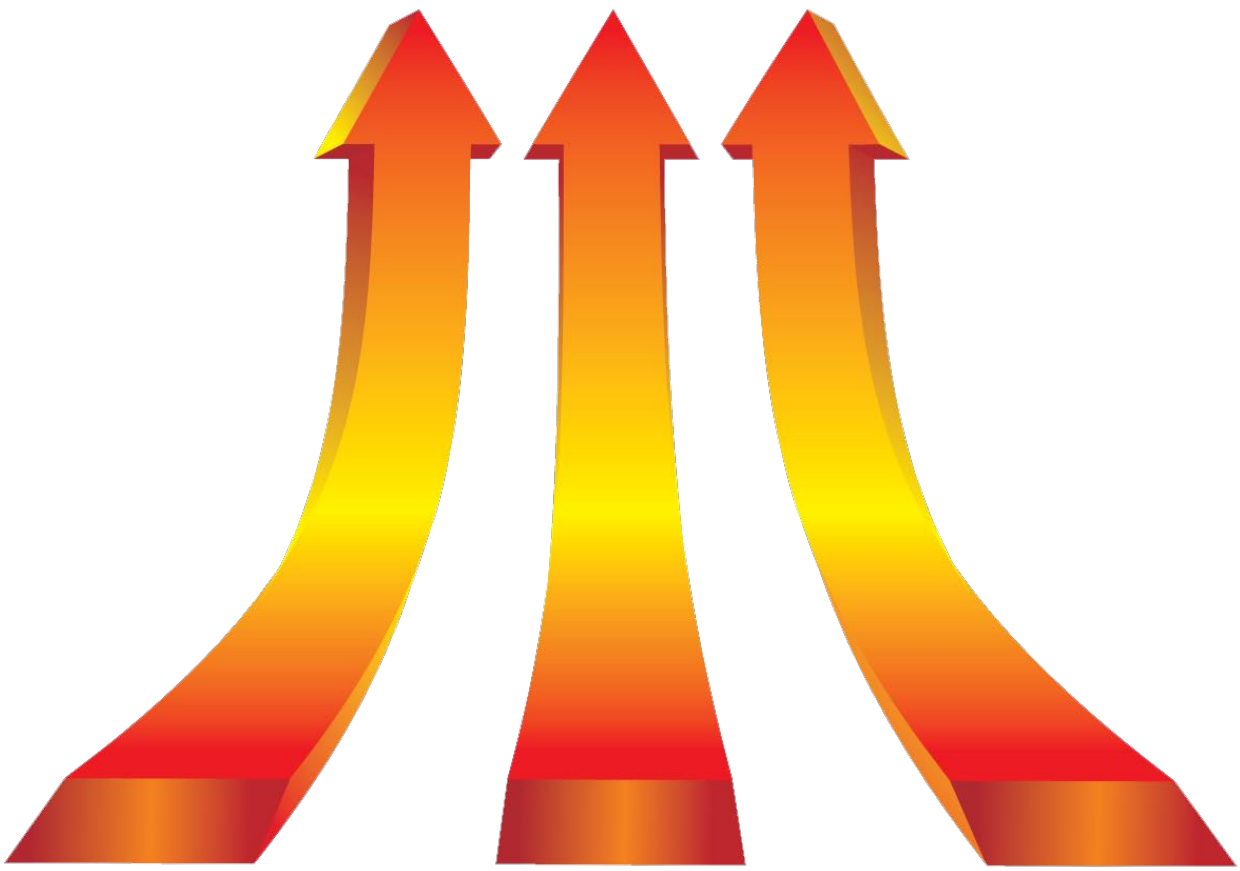
Paulo Quicoli
pauloquicoli@gmail.com
Editor geral da revista ClubeDelphi e editor técnico da .Net Magazine. Formado em processamento de dados pela FATEC-TQ. Atua como analista programador de sistemas e coordenador técnico da equipe de desenvolvimento .NET da Siplan Control-M (www.siplancontrolm.com.br) unidade Jaboticabal, e professor da FATEC, campus Taquaritinga. Blog: http://quicoli.worpress.com

Dê seu feedback sobre esta edição!

A Clubedelphi tem que ser feita ao seu gosto. Para isso, precisamos saber o que você, leitor, acha da revista!

Dê seu voto sobre este artigo, através do link:

www.devmedia.com.br/clubedelphi/feedback

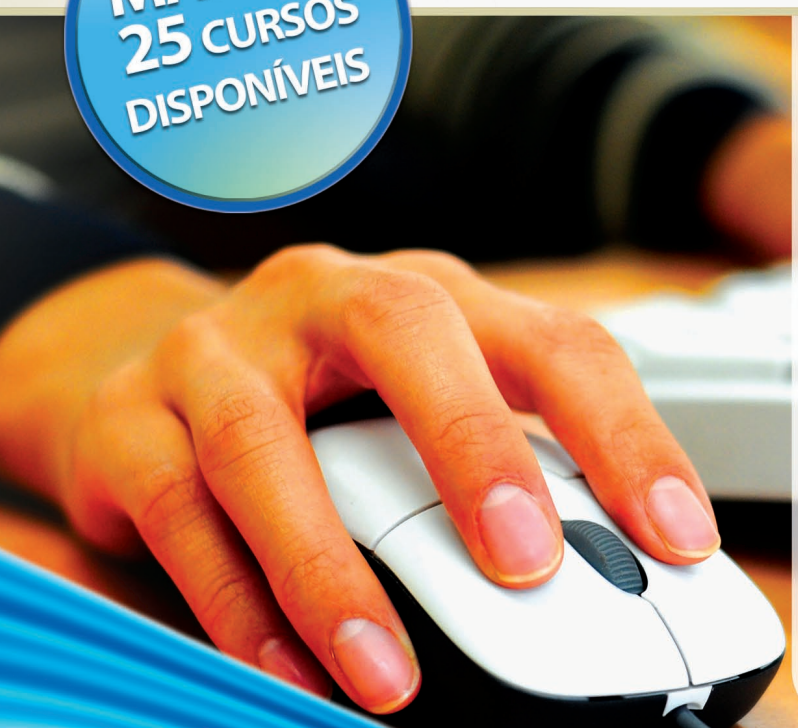


CURSOS ONLINE

.net magazine

A Revista .net Magazine oferece aos seus assinantes uma série de Cursos Online de alto padrão de qualidade.

MAIS de 25 CURSOS DISPONÍVEIS



CONHEÇA OS CURSOS MAIS RECENTES:

- Curso .net Entity Framework - Loja Virtual
- Novidades do Visual Studio 2010
- Componentes DevExpress
- Crie um chat utilizando Asp.Net 3.5 e Linq
- Novidades do C# (csharp) 3.0
- POO com C#
- Microsoft Visual Studio Tools for Office (VSTO)
- Computação na Nuvem com a Plataforma Microsoft Windows Azure

Para mais informações :

www.devmedia.com.br/curso/netmagazine

(21) 3382-5038

http://www.devmedia.com.br

DevMedia

TENHA ACESSO ILIMITADO A TODO CONTEÚDO DO SITE DEVMEDIA

SEJA UM ASSINANTE MVP!

Tenha acesso garantido a qualquer tema que você possa necessitar!

VIDEOS -AULAS

O assinante MVP acessa todas as vídeo-aulas do site DevMedia. Nenhum outro produto dá essa possibilidade. São mais de **7.000 video-aulas.**

CURSOS

Nosso site possui atualmente **+ de 250 cursos online** com a qualidade que você já conhece e confia. Com o MVP você terá acesso a todos eles.

REVISTAS

Assinante MVP acessa todas as nossas revistas na versão digital, incluindo todas as edições anteriores. Quando pintar aquela dúvida você terá mais de **3.500 artigos para consultar!**

ASSINATURA MVP MOST VALUABLE PROFESSIONAL

A assinatura MVP te dá a oportunidade de aumentar seu conhecimento em diversos assuntos, com qualidade e economia.

Invista na sua carreira se torne um profissional ainda mais completo!

POR APENAS R$59,90 POR MÊS

DevMedia

Acesse: www.devmedia.com.br/mvp